Faculdade de Engenharia da Universidade do Porto

Análise e Desenvolvimento de Sistema de Informação Fabril Orientado ao Processo

Roberto Carlos Barradas de Sousa

Dissertação realizada no âmbito do
Mestrado em Automação, Instrumentação e Controlo

Orientador: Prof. Dr. José António Rodrigues Pereira de Faria

Setembro de 2009

Resumo

A recolha e tratamento automático de dados de produção é um problema enfrentado por grande parte das empresas industriais, devido ao esforço de melhoria contínua que são obrigadas a desenvolver para acompanhar os desafios competitivos a que estão sujeitas.

Frequentemente, as empresas verificam que o seu parque de máquinas não foi desenvolvido tendo presente essa necessidade, dispondo de máquinas sem qualquer possibilidade de comunicação com o exterior, baseadas em tecnologias diferentes de controlo e provenientes de diferentes fabricantes de hardware com tecnologias incompatíveis entre si.

Existem no mercado inúmeros sistemas para ajudar na gestão e controlo da área de produção das empresas, mas tendem a ser aplicações isoladas ou, pelo menos, com limitada integração entre si. Os seus protocolos de comunicação proprietários tornam os sistemas de análise desconexos dentro da empresa e influenciam as futuras escolhas para garantir a total compatibilidade entre os sistemas.

Esta dissertação apresenta os procedimentos e análises efectuados para desenvolver um sistema de informação orientado para os processos fabris, capaz de recolher automaticamente dados provenientes de um conjunto heterogéneo de máquinas e equipamentos de fabrico.

Ao se iniciar projectos deste tipo, é necessário ter referências fundamentadas que permitam ver antecipadamente os caminhos que podem levar os projectos a bom porto. É nesse sentido que, neste trabalho, serão apresentadas referências para o desenvolvimento e compatibilidade de integração de sistemas, como sejam as tecnologias OPC, COM e DCOM.

Tendo como foco principal a tecnologia OPC, este trabalho faz também referência a componentes de software desenvolvidos em COM e DCOM, e efectua um estudo sobre o desempenho de um cliente OPC.

Palavras-chave: integração de sistemas, componentes de software, COM, DCOM, OPC, OPC DA, servidor, cliente, cliente OPC, *driver*, controlador, especificação, protocolos de comunicação.

Versão 1.0 (8 de Setembro de 2009)

Abstrat

The collection and automatic treatment of production data are a problem faced for a great part of the industrial companies, as a consequence of the effort of continuous improvement that they are obliged to develop in order to follow the competitive challenges that they are faced with nowadays.

Frequently, the companies verify that its park of machines was not developed having present this necessity, as the machines have no communication with the exterior, different technologies of control, manufacturers of the hardware with incompatible technologies and machines of different marks or manufacturers.

There are in the market innumerable systems to help in the management and control of the areas of production of the companies, but they tend to be applications isolated one from the others or at least with little integration between them. The protocols of communication proprietors make the systems of analysis disconnected inside the company and influence the future choices to guarantee the total compatibility between the systems.

This study presents the procedures and analyses accomplished to develop a system of information guided for the manufacture processes, capable to collect automatically given proceeding from a heterogeneous set of machines and equipment of production.

When initiating this type of projects, it is necessary to have based references that allow anticipatedly to see which ways can be better or more effective. It is in this direction that, in this work, will be presented references for the development and compatibility of integration of systems, as they are technologies OPC, COM and DCOM.

Having as main focus OPC technology, this work also makes reference to the components of Software developed in COM and DCOM, and it studies the performance of a client OPC.

Key-Words: integration of systems, components of software, COM, DCOM, OPC, OPC DA, server, client, client OPC, driver, controller, specification, protocols of communication.

Versão 1.0 (8 de Setembro de 2009)

Agradecimentos

À Sara e à pequena Maísa.

Ao pai Manuel, à mãe Maria e à mana Sofia.

Ao meu orientador, Professor José Faria, que pelo incentivo, crítica construtiva e constante disponibilidade, permitiram a concretização do trabalho apresentado.

Versão 1.0 (8 de Setembro de 2009)

“A essência do conhecimento consiste em aplicá-lo, uma vez possuído".

Confúcio

“A verdade de um curso não está no que aí se aprende, mas no que disso sobeja: o halo que isso transcende e onde podemos achar-nos homens.”

Virgílio Ferreira

Versão 1.0 (8 de Setembro de 2009)

Conteúdo

Versão 1.0 (8 de Setembro de 2009)

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

ADO	Activex Data Object
CLSID	Class Identifier
COM	Component Object Model
DCE	Distributed Computing Environment
DCOM	Distributed Component Object Model
DDE	Dynamic Data Exchange
DLL	Biblioteca de Software
DSN	Data Source Name
ERP	Enterprise Resource Planning
GUID	Global Unique Identifier
HMI	Human Machine Interface
HTTP	Hypertext Transfer Protocol
IID	Interface Identifier
IPC	Computador Industrial
LIMS	Laboratory Information Management Systems
MES	Manufacturing Execution System
ODBC	Open Data Base Connectivity
OEE	Overall Equipment Efficiency
OLE	Object Linking and Embedding
OPC	Open Process Control
OPC AE	Especificação Alarms & Events
OPC DA	Especificação Data Access
OPC HDA	Especificação Historical Data Access
OPC UA	Especificação Unified Arquitectura
OLEDB	Object Linking and Embedding, Database
RPC	Remote Procedure Call

SI	Sistema de Informação
SOAP	Simple Object Access Protocol
SCADAS	Supervisory Control and Data Aquisition
TCP/IP	Transmission Control Protocol / Internet Protocol
VB	Visual Basic
XML	eXtensible Markup Language
.NET	Tecnologia de Programação da Microsoft

Capítulo 1

Introdução

O trabalho apresentado neste documento foi desenvolvido no âmbito do Mestrado em Automação, Instrumentação e Controlo.

Este primeiro capítulo começa por apresentar a motivação, os objectivos e o contexto em que decorreu o trabalho.

É também descrita a metodologia seguida e a organização do documento.

1.1 Motivação

Na base do sistema aqui apresentado está a necessidade de recolha automática de informação sobre os processos de fabrico que permita, numa fase posterior, proceder à avaliação do desempenho e ao desenvolvimento do sistema produtivo.

A recolha e tratamento automático de dados de produção é um problema enfrentado por grande parte das empresas industriais, devido ao esforço de melhoria contínua que são obrigadas a desenvolver para acompanhar os desafios competitivos a que estão sujeitas actualmente.

O tradicional registo dos dados pelo operador em papel, ou mesmo directamente em suporte informático, é enfadonho, moroso, susceptível de erros e muitas vezes não permite conhecer, em tempo útil, o evoluir do processo produtivo. Neste contexto, podem ser consideradas vários tipos de empresas, conforme o grau de automatização na recolha e no tratamento dos dados de produção. O estudo aqui apresentado tem por alvo as empresas com um elevado grau de automatização dos processos físicos de produção, mas que não dispõem de sistemas de informação que lhes permitam acompanhar em tempo real o desempenho desses processos.

Muito frequentemente estas empresas verificam que o seu parque de máquinas não foi desenvolvido tendo presente a necessidade de recolha automática de dados devido a factores como:

- Máquinas sem qualquer comunicação com o exterior
- Tecnologias diferentes de controlo
- Marcas de hardware com tecnologias incompatíveis
- Máquinas de diferentes marcas ou fabricantes
- ….

Nestas circunstâncias, o desenvolvimento de um sistema de informação, que permita recolher e tratar os dados é um problema complexo, em que é preciso considerar várias dimensões, conforme se demonstra ao longo deste documento.

1.2 Objectivos

Este projecto teve, como objectivos gerais, a análise e o desenvolvimento de um sistema de informação orientado para os processos fabris, capaz de (i) recolher automaticamente dados provenientes de um conjunto heterogéneo de máquinas e equipamentos de fabrico, e (ii) calcular e apresentar indicadores de desempenho relativos à eficiência dos equipamentos da linha de montagem, tais como OEE, tempo de paragem e respectiva causa, tempo para ajustes, faltas de material, tempo de manutenção.

Com o decorrer do projecto os objectivos centraram-se mais especificamente nos aspectos relacionados com a recolha e comunicação de dados entre os equipamentos físicos de produção e o servidor onde reside a base de dados e a aplicação de interface com o utilizador, nomeadamente:

- no desenvolvimento de uma arquitectura que implemente uma solução de comunicações comum ao conjunto de equipamentos de produção, que seja flexível e que permita futuros desenvolvimentos.
- na análise e aplicação das tecnologias COM, DCOM e, em especial, da tecnologia OPC.
- na análise e avaliação do desempenho da tecnologia OPC.
- no desenvolvimento e teste da aplicação de aquisição de dados em condições reais de utilização.

1.3 Enquadramento do Projecto

O projecto aqui apresentado teve como motivação directa uma necessidade da empresa *Huf* Portuguesa (Fábrica de Componentes para o Automóvel), situada em Tondela. A empresa possui um parque de máquinas vasto e heterogéneo, apresenta um elevado nível de automatização dos processos com recurso a inúmeras máquinas, provenientes de vários fabricantes e integradores e baseadas em diferentes tecnologias e tipos de controlo.

A *Huf* possui já um sistema de informação que recolhe os dados referentes ao produto, mas não possui um sistema de informação que lhe permita conhecer o desempenho dos equipamentos, ou seja do processo fabril. Também não dispõe de registos, em papel ou em suporte digital, que permitam quantificar o desempenho da grande variedade e número de equipamentos que possui.

A necessidade continua de aumentar a capacidade produtiva, a contabilização e consequente redução de custos, a obtenção de uma eficiência global superior e um aproveitamento mais eficiente dos equipamentos, levaram a que a empresa decidisse implementar um sistema de informação que lhe permitisse conhecer a eficiência dos seus equipamentos.

Existem no mercado várias tecnologias e aplicações passíveis de ser utilizadas na implementação de sistemas deste tipo que, por vezes, se complementam, mas que a grande parte das vezes são incompatíveis entre si. Ao se iniciar projectos deste tipo, é necessário ter referências fundamentadas que permitam ver antecipadamente quais os caminhos que podem levar os projectos a bom porto. É nesse sentido que, neste trabalho, serão apresentadas referências para o desenvolvimento de projectos a este nível, como sejam as tecnologias OPC, COM e DCOM.

1.4 Metodologia

A metodologia seguida no desenvolvimento do projecto envolveu as seguintes etapas:

Numa primeira fase, efectuou-se um estudo ao sistema de produção da empresa com o objectivo de (i) conhecer as necessidades da empresa, (ii) conhecer as tecnologias existentes e os sistemas de controlo utilizados, (iii) seleccionar as máquinas representativas para realizarem os testes do projecto, (iv) compreender e registar os indicadores mais importantes para a *Huf*, (v) identificar quais os dados a recolher para calcular e apresentar os respectivos indicadores.

De seguida, foi realizado um estudo sobre as tecnologias de comunicação industrial, o qual permitiu definir quais as tecnologias mais adequadas para aplicar no projecto.

Procedeu-se então ao estudo das tecnologias COM, DCOM e OPC e à avaliação do seu desempenho face aos requisitos de comunicação do projecto.

Por fim, foi desenvolvido e testado o módulo de aquisição de dados, com especial relevância no cliente OPC DA, em condições reais de funcionamento.

1.5 Estrutura da Dissertação

Para além da introdução, esta dissertação contém mais 7 capítulos:

No capítulo 2, é realizada uma análise de requisitos para ao sistema de informação.

No capítulo 3, é descrita a concepção do sistema de informação.

No capítulo 4, é realizado um estudo sobre COM e DCOM, seguido da apresentação de um cliente e servidor.

No capítulo 5, é realizado um estudo sobre OPC e a especificação OPC DA

No capítulo 6, é apresentado o cliente OPC com acesso a base de dados, desenvolvido de raiz para realização de testes de desempenho relativos à especificação OPC DA.

No capítulo 7, são explicados os constituintes do sistema de informação e descritos os passos para a implementação do modulo de aquisição de dados.

No capítulo 8, são apresentadas as conclusões e trabalho futuro

Capítulo 2

Análise de Requisitos para o Sistema de Informação

Neste capítulo começar-se-á por apresentar as necessidades e problemas detectados pela empresa. Seguidamente, apresentamos as análises efectuadas relativamente ao sistema de produção e às máquinas escolhidas para o estudo. Para finalizar, cruzamos os dados obtidos nas análises com os objectivos da dissertação e definimos os requisitos e os indicadores que o sistema de informação deve apresentar.

2.1 Necessidades da Empresa em Estudo

Numa fase inicial do projecto, foi efectuada uma reunião com a equipa de Engenharia da *Huf* Portuguesa com o objectivo de recolher quais as necessidades e os problemas identificados pela empresa. Nessa reunião, a *Huf* exprimiu as seguintes necessidades:

- Dispor de um sistema de informação orientado ao processo fabril que permitisse conhecer a eficiência dos equipamentos de fabrico.
- A solução a desenvolver deve ser baseada em soluções standard e na utilização da rede *ethernet* já existente
- A solução a desenvolver deve ser flexível de forma a permitir futuras evoluções.
- Interligar os dispositivos da linha de montagem a um computador central capaz de reunir a informação da linha numa base de dados.
- O sistema a desenvolver deve fornecer indicadores de desempenho e de eficiência, tais como *stop time*, *tack time*, *repair time, OEE, bottleneck station*, erros dos operadores,

apresentar estatísticas entre outros (Os indicadores de desempenho serão explicados no ponto 2.4).

De entre as dificuldades detectadas pela *Huf*, salientavam-se a heterogeneidade dos equipamentos quer ao nível do tipo de máquinas, quer ao nível do controlo e a dificuldade de como e onde recolher os dados que nos permitissem o calculo dos indicadores mais complexos.

Partindo dos dados conhecidos no momento, foi elaborado um plano de trabalho que, numa fase inicial, consistiu na realização de várias visitas técnicas para recolher no terreno dados que permitissem caracterizar o sistema de produção em estudo.

2.2 Análise do Sistema de Produção

As visitas técnicas realizadas permitiram efectuar uma análise ao sistema de produção da empresa e retirar um conjunto de conclusões, apresentadas de seguida, e que determinaram em larga medida o caminho a seguir (Para uma melhor compreensão das observações que serão descritas neste tópico deve ser lido o anexo A).

A análise revelou que as seis linhas de montagem eram similares, por essa razão iríamos concentrar o estudo em apenas uma linha. Analisando então uma das linhas, verificou-se que, em relação ao tipo de controlo, existiam dois grandes tipos de máquinas, as máquinas controladas por PLCs (Autómatos) e as máquinas controladas por IPCs (Computadores Industriais).

No vasto e heterogéneo número de máquinas da linha de montagem (30 máquinas), podia-se encontrar cinco marcas diferentes de PLCs e vários fabricantes e integradores, o que levou a que o estudo numa primeira fase fosse limitado a uma máquina representativa para cada tipo de controlo.

Deste estudo também ficou claro que a principal questão a resolver seria a comunicação com os equipamentos de fabrico, por duas razões essenciais que se prendiam pela heterogeneidade e pelo facto de os fabricantes e integradores não terem previsto soluções que permitissem a comunicação com o exterior do equipamento. Perante as conclusões anteriores verificou-se que os esforços a desenvolver seriam maioritariamente canalizados dentro da temática das comunicações industriais.

Um aspecto relevante a ter em conta é a possibilidade de acesso ao código da programação de cada máquina. Esta empresa dispõe de código aberto em praticamente todas as máquinas, porque estas foram desenvolvidas por encomenda, mas existem duas máquinas na linha que não permitem o acesso ao código. Nessas situações, será necessário contactar directamente os respectivos fabricantes para efectuarem as alterações necessárias, situação que

não se apresenta problemática pois a empresa mantêm contacto com todos os fabricantes e integradores.

No momento de seleccionar as máquinas que seriam objecto do estudo, foi tido em consideração que as máquinas escolhidas tinham de pertencer a integradores/fabricantes mais representativos na linha e serem dentro do seu grupo de controlo, aquelas que menos estariam preparadas para as alterações a realizar. Teve-se especial atenção para garantir que na linha não havia máquinas que por qualquer motivo ficassem de fora dos grupos de controlo observados.

Face ao verificado foram seleccionadas a máquina de Comprovar de Portas (*PSA*) e a Máquina de Linguetas. Seguidamente será apresentada a análise de cada um destes equipamentos.

2.3 Análise dos Equipamentos de Produção

A análise dos dois equipamentos seleccionados teve por objectivo conhecer o seu funcionamento, a sua constituição e o seu programa de controlo. Estes dados recolhidos são cruciais para definir o sistema de informação, uma vez que estão na base da sua arquitectura.

De notar que os elementos aqui apresentados foram recolhidos através de uma observação directa das máquinas, uma vez que os respectivos manuais se limitam a apresentar as máquinas e a descrever as instruções de funcionamento e configuração.

2.3.1 Máquina de Comprovar Portas

Trata-se de uma máquina que está inserida na linha que produz para o grupo *PSA* e que está representada na figura 2.1 Esta máquina efectua o teste das fechaduras direita e esquerda do automóvel. Esse teste consiste em recolher da palete a fechadura já montada e em executar vários testes que, com elevada segurança, permitem garantir que a mesma está operacional, ou seja bem montada e sem qualquer encravamento ou defeito.

Figura 2.1 – Máquina de comprovar portas

A máquina é controlada por um autómato *Omron Sysmac CJ1M / CPU21*, com três cartas de entradas e duas cartas de saídas para sinais digitais, um manipulador controlado por um *Servomotor Omron* que realiza o teste à fechadura, e uma consola táctil (*Beijer Electrocnics cimrex 69*), que permite:

- Comandar a máquina.
- Acompanhar a sequência do ciclo de teste.
- Afinar a máquina.
- Saber onde ocorreu uma falha.
- Apresentar instruções de operação e em caso de falha.

A sequência de funcionamento para o ciclo normal de teste com manipulador é a seguinte e, conforme, referido, foi obtida recorrendo a uma observação directa da operação da máquina:

Para entrar em serviço, é premido o botão *Start*, após o que a máquina realiza o seguinte ciclo de operações:

- Detectar presença de palete no *stopper*
- Elevador tranca a palete
- Manipulador baixa pinça
- Manipulador fecha pinça
- Manipulador sobe pinça
- Manipulador recua
- Manipulador baixa pinça
- Tranca esquerda e direita trancam a peça

- Manipulador abre pinça
- Manipulador recua
- Máquina executa o teste da peça
- Manipulador retira a peça e coloca-a na palete
- Elevador baixa a palete
- *Stopper* liberta palete

No terminal de interface com o operador, apenas são visualizados os seguintes estados (sequência retirada também de observação directa):

Inicio
1- *Start*
Ciclo de trabalho (repetitivo)
2-Maq. em espera da chegada da palete
3-Ciclo de alimentação da peça
4-Ciclo de teste
5- Ciclo Retirar a peça

Uma falha pode ocorrer em qualquer estado do ciclo normal de teste. Perante uma situação de falha, a sequência de operações é a seguinte:

A – Ocorrência da falha (evento erro)
As falhas possíveis estão catalogadas, estando incluídas nas falhas, as ocorrências de peças não conformes.
Na sequência da ocorrência de uma falha, o respectivo código é visualizado no terminal de interface com o operador (*touch screen*), após o que o operador ou o responsável pela manutenção a reconhece e a corrige.

B – Reconhecimento do erro
O operador prime o botão “*reset*” para reconhecer o erro.

C – Reinício do ciclo de trabalho
Operador prime no botão “*Start*” (evento *start*) , terminando a sequência da falha e passando a máquina ao estado “*Ready*”, i.e., pronta para entrar num novo ciclo.

Sempre que ocorre uma anomalia aparece uma mensagem de erro na consola, a tabela do Anexo B apresenta as possíveis anomalias detectáveis pela máquina.

A informação do tipo do erro é crucial que, conforme se verá mais à frente, será associada aos dados a transferir e a arquivar na base de dados, ajudando deste modo na construção dos indicadores de desempenho.

Embora fosse possível aceder ao programa do autómato, não havia acesso aos comentários nem ao nome das variáveis e muito menos a uma descrição da sua utilização. Apenas tivemos acesso ao *Grafcet* geral representado na figura 2.3 o qual, no entanto, estava incompleto e apenas permitiu conhecer os estados principais do programa (que podiam, aliás, ser conhecidos directamente a partir da observação do funcionamento da máquina).

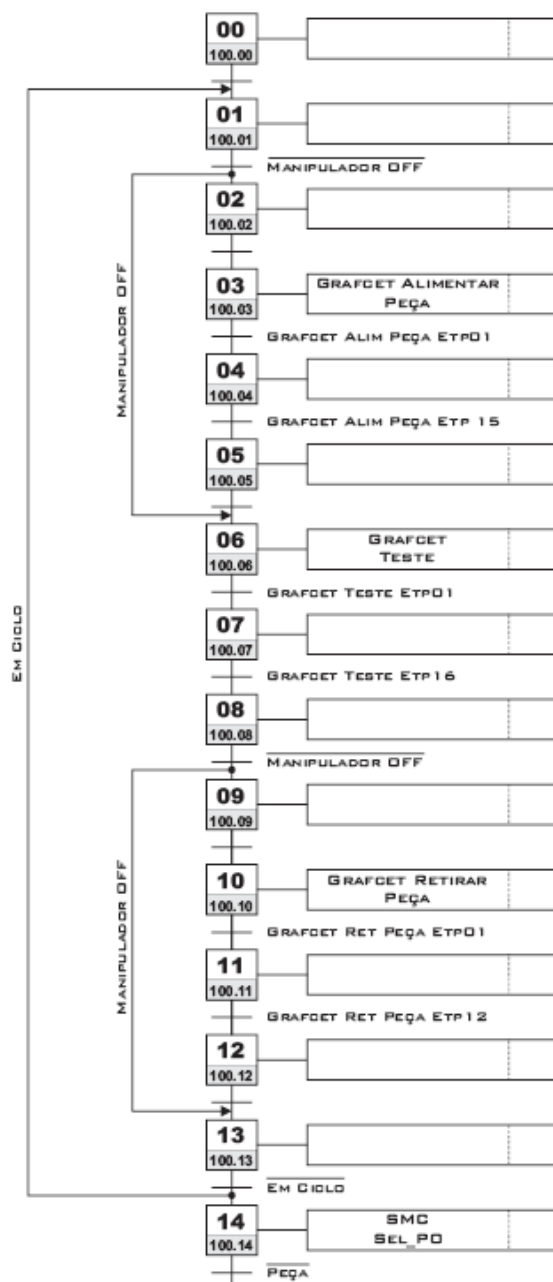



Figura 2.2 – Grafcet Geral da máquina

Foi também pedido ao Integrador para disponibilizar o programa comentado mas, alegando tratar-se de informação confidencial sobre o manipulador e o controlo do servomotor, isso não foi possível.

Em consequência, foi necessário executar um estudo minucioso do programa do autómato que permitiu construir o seguinte diagrama de estados, que indica os estados e os eventos principais. Este diagrama está relacionado, e que funciona sobre com o *Grafcet* geral, mas retém apenas os "eventos relevantes" do ponto de vista da monitorização da máquina. Um evento relevante é, por exemplo, o início de ciclo. Já o fecho de uma electroválvula no meio do ciclo não é considerado um evento relevante, a menos que essa electroválvula tenha encravado.

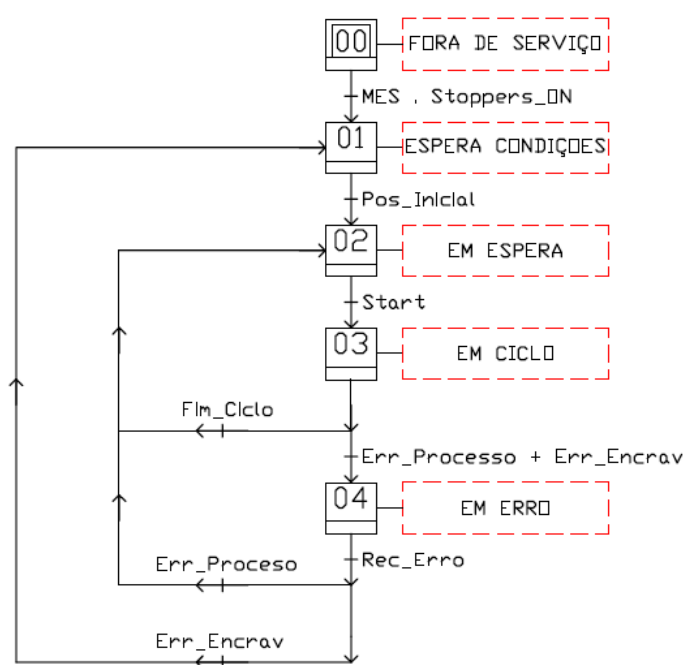



Figura 2.3 – Diagrama de estados obtido

De acordo com este diagrama, do ponto de vista da monitorização, a máquina tem 5 estados relevantes, a saber:

00 – "Fora de Serviço" a máquina encontra-se ligada a corrente mas espera que os *Stoppers* (parte da máquina onde são recebidas as paletes) estejam activados (evento Mes.Stoppers_ON) para receberem as paletes com as peças, passando assim para o próximo estado.

01 – "Espera de condições" a máquina está ligada com os *Stoppers* ligados mas espera ter condições para poder receber peças. Razões para ter condições são por exemplo falta de configuração para a peça a executar, o manipulador não está na posição correcta para efectuar a operação ou qualquer encravamento de ordem mecânica. No momento que estiverem reunidas

as condições e a máquina esteja na posição inicial (Evento Pos_Inicial), passamos ao próximo estado.

02 – “Em Espera” a máquina está ligada com os *Stoppers* ligados, na posição inicial, quando se dá o evento *Start*, passa ao próximo estado. O evento start só está activo quando se verificarem duas condições, a primeira é que o botão de “*Start*” esteja activado e a segunda é que exista uma palete no *Stopper*

03 – “Em ciclo” verificaram-se todas as condições anteriores e a máquina está a operar o seu ciclo de funcionamento. Nesta posição temos duas possibilidades, se o ciclo ocorrer com normalidade dá-se o evento “Fim_Ciclo” e a máquina volta ao estado “Em Espera”, mas se surgir algo de anormal (ocorrer o evento Err_Processo ou evento Err_encravamento) a máquina passa ao estado “Em Erro”.

04 – “Em Erro” quando se entra neste estado o operador é informado do tipo de erro pela consola táctil, têm que reconhecer e corrigir o erro (carrega no botão de reconhecimento do erro “evento Rec_Erro”). A partir deste ponto consoante o tipo de erro temos caminhos diferentes, se for um erro de encravamento da a máquina passa para o estado “Espera Condições” mas se for um erro de processo passa para o estado “Em Espera”.

2.3.2 Máquina de Linguetas

Trata-se de uma máquina que está inserida na linha que produz sistemas de fechaduras para o grupo automóvel *PSA*(*Peugeot e Citroën*). A máquina insere as linguetas e molas nos cilindros das fechaduras, que permitem a codificação da fechadura, para que esta apenas abra com a chave correspondente.

Figura 2.4 – Imagem de um cilindro, linguetas e molas.

No final do ciclo, a máquina realiza um teste para verificar se a fechadura está bem montada e operacional.

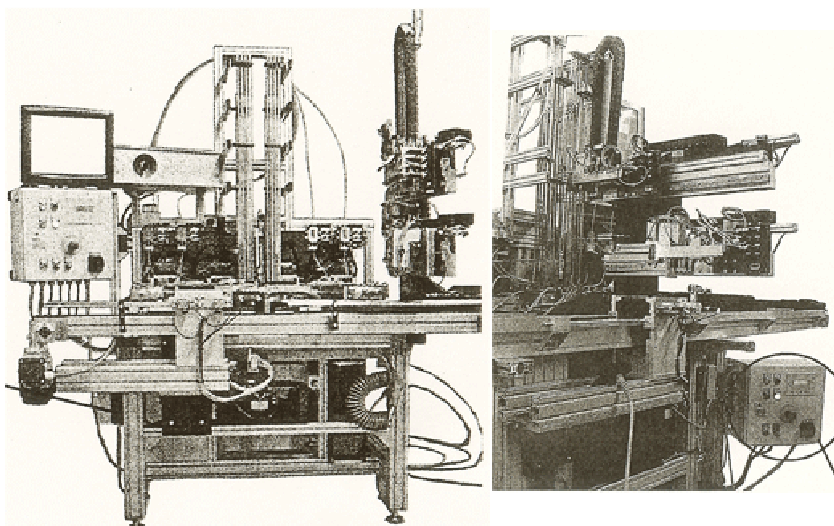



Figura 2.5 – Máquina de Linguetas

A máquina é controlada por unidade de comando que é constituída por um PC industrial (IPC) com processador *Pentium IV* a 2,4Ghz e que possui as seguintes cartas de comunicação:

- Cartas para sinais analógicos (controlo do apalpador de molas).
- Cartas *Can-bus* (para o controlo dos *drivers* dos motores).
- Cartas *AS-I* (para comunicar como os autómatos)
- Carta *ethernet* (para comunicação com o exterior)

Outros equipamentos de controlo relevantes na máquina de linguetas são os autómatos, *Siemens Simatic S7-300* e o autómato *Festo FC640,* que são os responsáveis pelo controlo do *Handling* (conjunto dos dois manipuladores da máquina).

A comunicação com o operador é realizada com recurso a um terminal táctil *Elo TouchsSystems*.

A aplicação de controlo no IPC está desenvolvida em C++ e não é de código aberto. Logo, qualquer mudança no software de controlo necessita do acordo e negociação com o fabricante. No entanto, e como já foi referido, há abertura por parte do fabricante para introduzir alterações na aplicação.

A sequência de funcionamento para o ciclo normal de teste com manipulador é a seguinte (a sequência seguinte foi obtida por observação directa da maquina):

Para entrar em serviço, é premido o botão *Start*, após o que a máquina realiza o seguinte ciclo de operações:

- Detectar presença de palete no *stopper*
- Elevador tranca a palete
- O *Handler* de cilindros retira os cilindros da palete

- O *Handling* de cilindros move/posiciona os cilindros
- O *Handling* de cilindros coloca os cilindros na máquina de montagem de cilindros
- É Inserida a mola
- São Inseridas as linguetas
- Faz o procedimento inverso (retira o cilindro e coloca-o na palete)
- Elevador baixa a palete
- *Stopper* liberta palete

As falhas também estão catalogadas, no entanto não estão disponibilizadas, assim como qualquer informação que nos permita conhecer ao pormenor o ciclo de funcionamento, resta-nos arranjar uma técnica que sem uma intervenção de fundo na aplicação de controlo seja capaz de nos fornecer os dados sobre a máquina.

Em caso de avarias da máquina como seja a aplicação deixa de funcionar, o sistema operativo cria um erro crítico ou o disco avaria, a solução passa por repor uma imagem do sistema.

2.4 Definição dos Requisitos e Indicadores

Como resultado da análise efectuada, e tendo presente os objectivos e as necessidades da empresa, foram definidos os requisitos e os objectivos gerais do sistema de informação a desenvolver. Assim, o sistema de informação deveria de ser capaz de:

- Monitorizar do processo de fabrico em tempo real.
- Recolher e registar os dados sobre a eficiência dos equipamentos.
- Fornecer Indicadores estatísticos.
- Permitir conhecer a contribuição de cada máquina para o desempenho do processo produtivo.
- Permitir a monitorização remota do processo e fabrico.

Relativamente à visualização do processo produtivo, o sistema devia apresentar:

- O Estado actual dos equipamentos sob a forma de um sinóptico do processo de fabrico.
- Os Indicadores, as estatísticas e gráficos que o sistema vai calcular e ou fornecer.
- Erros ocorridos nas máquinas
- Relatórios e históricos sobre a produção, erros no parque das máquinas.
- Disponibilizar os dados em folha de cálculo.

Relativamente à avaliação do desempenho, pretendia-se que o sistema fornecesse os seguintes indicadores:

- Disponibilidade.
- *Stop time* (tempo de paragem), indicador que contabiliza o tempo em que a máquina esteve parada.
- *Tack time* (tempo de operação), indicador que contabiliza o tempo em que a máquina esteve em operação
- Repair time (tempo de reparação), indicador que contabiliza o tempo em que a máquina esteve em reparação ou manutenção
- OEE (eficiência geral do equipamento) indicador que apresenta a eficiência do processo de fabrico.
- *Bottleneck station* (gargalo da estação) ponto do sistema que condiciona a produção, porque o desempenho do sistema depende deste ponto
- Taxa de não conformidade
- Listagem de micro paragens
- Erros dos operadores
- Outros que se verifiquem necessários

Os indicadores do sistema permitem os seguintes benefícios:

- Melhorar a eficiência global do sistema.
- Permitir detectar mais rapidamente esforços de melhoria.
- Permitir responsabilizar e reclamar com os integradores.
- Introduzir requisitos de eficiência operacionais na aquisição de novos equipamentos.
- Criação de um projecto standard que seja flexível e que permita futuros desenvolvimentos, por outras palavras permita evoluir e adaptar-se a novas exigências.

2.5 Sintese

No final deste capítulo passamos a: (i) conhecer as necessidades e as dificuldades da empresa, na tentativa de implementação de um sistema de informação orientado ao processo de fabrico. (ii) conhecer o funcionamento e a constituição dos equipamentos escolhidos para realização do projecto, os respectivos diagramas de estados relevantes para a monitorização do desempenho. (iii) ter definidos os requisitos para o sistema de informação, entre os quais os indicadores de desempenho que deve ser capaz de apresentar.

Capítulo 3

Concepção do Sistema de Informação

Neste capítulo é analisada a arquitectura do sistema de informação fabril, onde o principal desafio consistiu em encontrar a solução mais eficiente para a recolha e comunicação da informação proveniente das máquinas.

Este é um problema complexo quando o parque de máquinas é vasto e heterogéneo, proveniente de vários fabricantes e integradores e apresenta diferentes tecnologias, tal como se verifica no caso em estudo.

Na definição da arquitectura foram consideradas 5 questões principais: (i) a arquitectura geral do sistema, (ii) a topologia da rede, (iii)o sistema operativo a escolher, (iv) a ferramenta de desenvolvimento, (v) a linguagem de programação em que será desenvolvido o sistema de informação. Nos itens seguintes vão ser analisados cada um destes tópicos e justificadas as opções tomadas. Na parte final do capítulo, são discutidas as ferramentas de programação a utilizar no desenvolvimento do sistema.

3.1 Arquitectura Geral do Sistema

Conforme já foi referido, a fábrica possui 6 linhas de produção idênticas. Para garantir a autonomia de cada linha ao nível da recolha de dados, optou-se por instalar um computador por linha e um computador central, conforme representado na figura seguinte.

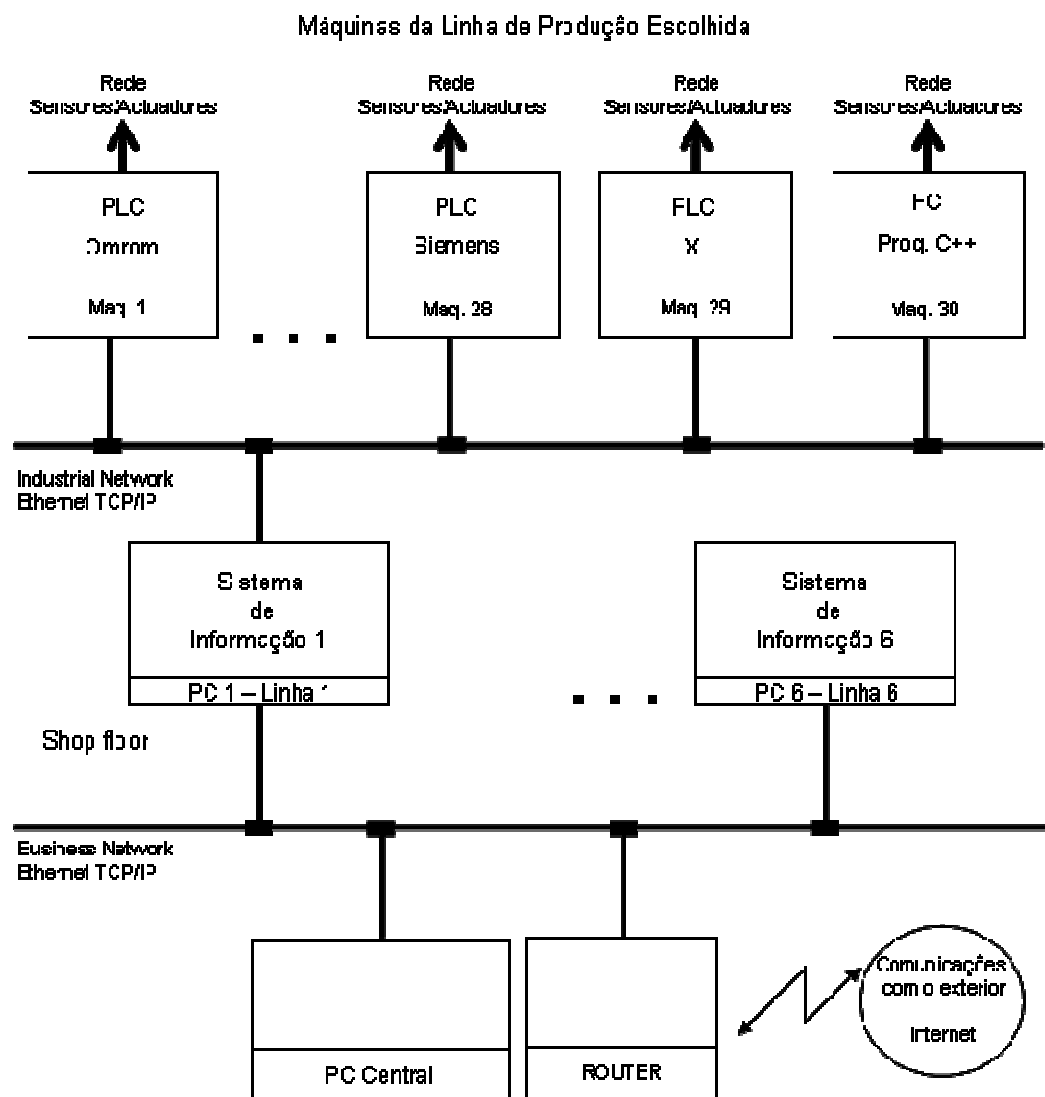



Figura 3.1 – Arquitectura geral do sistema

A justificação principal para esta solução prende-se com o facto do volume de dados a recolher, em cada linha, ser relativamente elevado. De facto, as operações nas máquinas demoram apenas alguns segundos (por exemplo, 8 segundos no caso da máquina de comprovar peças) o que dá origem a um grande volume de dados a transmitir. Esta opção foi também reforçada pelo facto de a empresa já dispor de um computador em cada linha produção e um central, que até ao momento, suportam o sistema de informação orientado ao produto.

3.2 Tipologia de Rede

Neste ponto é tratada outro aspecto fundamental da arquitectura do sistema de informação, o qual se prende com a tipologia da rede de comunicações.

Analisando as tecnologias disponíveis no mercado, concluiu-se que havia duas alternativas principais a considerar neste projecto quanto à tipologia da rede de comunicações: a utilização de uma rede única, ou a implementação de um conjunto de redes proprietárias.

No primeiro caso, trata-se de criar uma rede que fosse suportada por todos os equipamentos existentes na fábrica, como é o caso da rede *ethernet*. A segunda alternativa consistia em recorrer a várias redes proprietárias em função dos diferentes protocolos utilizados pelo fabricante de cada máquina.

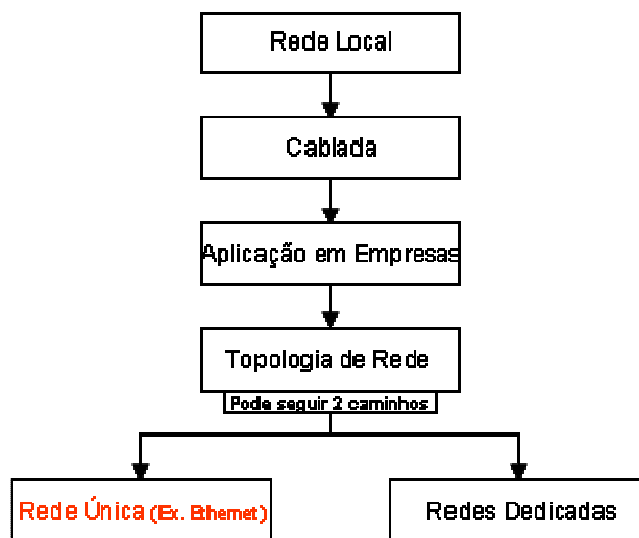



Figura 3.2 – Soluções a considerar na escolha da tipologia da rede

A opção por redes proprietárias, em projectos deste tipo, apenas é viável quando o projecto é desenvolvido de raiz, desde que, ainda na fase de projecto, haja o cuidado de garantir que todas as máquinas utilizam o mesmo protocolo.

No caso deste projecto, como as máquinas foram desenvolvidas em momento de tempo distintos e por vários integradores, sem ter presente o pressuposto da comunicação com um sistema central exterior às linhas de produção, seria necessário a implementação de várias redes.

Nestas condições, a opção por uma rede única foi a solução encontrada, justificando-se esta opção, por razões económicas e logísticas. De facto, além de mais económica, esta solução ocupa menos espaço (onde por vezes este já é escasso),

Convêm, no entanto, referir que as principais vantagens das redes proprietárias, entre as quais o melhor desempenho e gestão de sistemas em tempo real, não são muito relevantes para o sistema de informação em estudo, uma vez que nesta aplicação projecto, conhecer o estado da linha com alguns segundos de atraso não é penalizador, além de que facilmente são desenvolvidos algoritmos que garantam o não congestionamento da rede, conforme se verá mais à frente.

3.3 Ferramenta de Desenvolvimento da Aplicação

As duas opções no ponto anterior, podem ser abordadas de duas formas distintas:

- ou desenvolvendo uma aplicação para o sistema de raiz, utilizando para isso linguagens e ferramentas de desenvolvimento de aplicações de uso genérico , por exemplo, programação em VB, .NET, C#, C++, Java…
- ou recorrendo a ferramentas de desenvolvimento dedicadas a aplicações deste tipo, tais como as existentes no mercado, da *Wonderware* (“*System Platform*”, “*In Touch HMI*”) ou da *Siemens* (“*WINCC*”, “*Simatic Net*”).

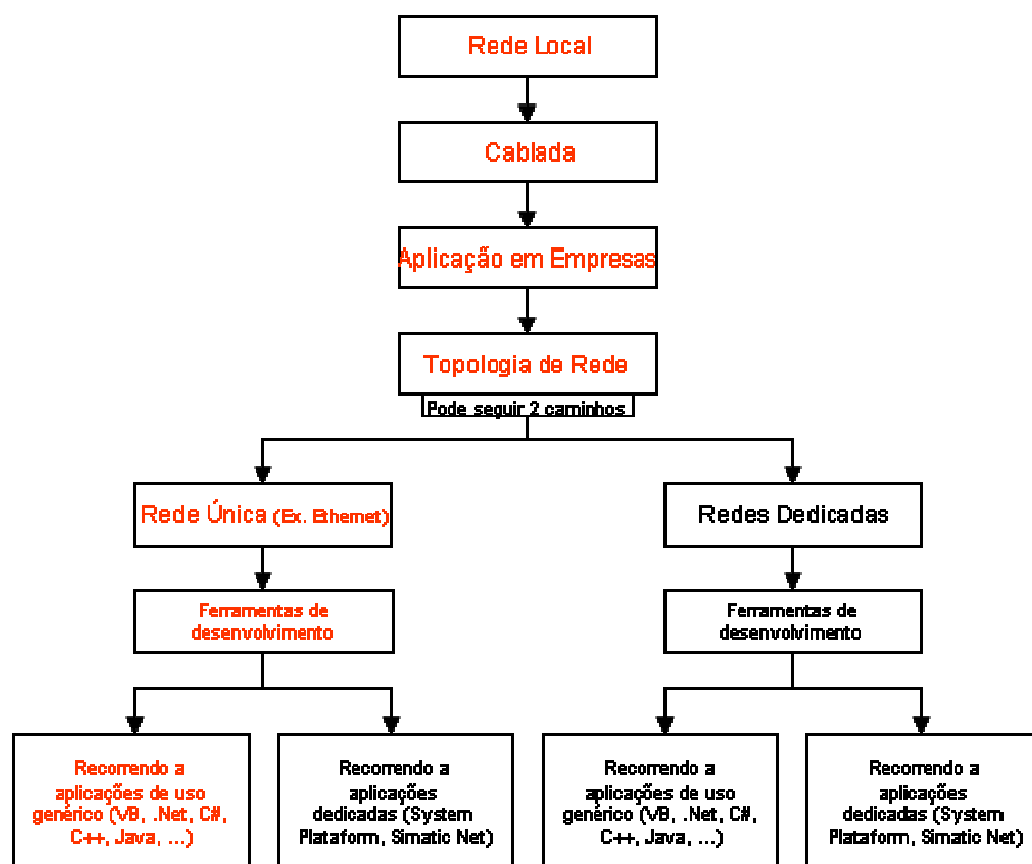



Figura 3.3 – Soluções a considerar na escolha da ferramenta de desenvolvimento

Sendo as duas abordagens viáveis à partida, a opção, vai depender da análise das vantagens de cada abordagem em conjugação com as valências que a empresa detém.
Nessa análise, podem ser considerados vários factores:

- Ordem financeira – talvez o mais importante, uma vez que o tecido empresarial quer ter o melhor sistema possível e que responda as suas necessidades ao melhor preço. Também não pode ser ignorado que um projecto muito caro pode levar a que a sua

viabilização seja impossível. Ter em consideração que uma solução que não possua os últimos avanços tecnológicos mas que seja robusta, eficaz e fiável pode ser economicamente a melhor solução para a empresa.

- Volume dados transferidos – É necessário conhecer o volume de dados a adquirir, registar e transferir, o qual depende do tamanho das mensagens e da cadência de transmissão, uma vez que é possível ter mensagens pequenas mas um elevado número de dados a registar no mesmo instante de tempo.

- Competências (*know-how*) da empresa - se a empresa tiver colaboradores com conhecimento para o desenvolvimento, teste e implementação do sistema de informação, ou se tem que recorrer a *outsourcing*.

- Ordem temporal – Se a empresa tiver uma grande necessidade de adquirir o sistema, pode levar a que se escolha a solução mais rápida de implementar. Como regra geral, a utilização de ferramentas de desenvolvimento de software em tempo real torna o desenvolvimento mais rápido, mas também mais dispendioso e por vezes menos flexível.

Conjugando os factores para a escolha da técnica com a realidade da empresa *Huf* verifica-se que:

- Relativamente aos factores económicos, é mais vantajoso a construção de uma só rede para a transferência de dados do que várias dedicadas em função dos protocolos utilizados pelos diferentes fabricantes presentes na fábrica em grande medida porque as cartas para redes proprietárias são, em geral, muito mais caras que cartas para *ethernet*, devido a grande implantação desta rede. É também importante referir que, em determinados equipamentos, não é possível adicionar cartas de comunicação por falta de *slots* de expansão ou mesmo por impossibilidade de receber qualquer carta. Em último recurso, esta limitação pode levar á utilização de um novo autómato programável. Nestes situações, a utilização de uma rede *ethernet* permite uma solução muito eficaz, que consiste na utilização da porta série receber um conversor *série-ethernet*, passando assim o dispositivo a comunicar com a rede.

- Quanto ao volume de dados, o projecto não têm um grande volume de dados a transferir, mas pode ter muitos acessos ao meio físico em simultâneo. As máquinas têm ciclos de funcionamento de 8 a 10 segundos, durante os quais tem de ser registados em

média 3 eventos. Como a linha têm 30 máquinas, teremos 700 registos aproximadamente por minuto. A situação crítica ocorre se todas as máquinas acedessem ao meio físico ao mesmo tempo. (situação que será analisada posteriormente).

- No que concerne ao terceiro factor, a empresa detêm algum conhecimento nesta área, mas necessita um acompanhamento com conhecimentos específicos e experiência para o desenvolvimento e teste do projecto. Aproveita deste modo as valências dos colaboradores da empresa e os recursos já existentes, motivo pelo qual sugere como requisito que, na recolha e transporte de informação se utilize a tecnologia *ethernet*. No entanto, como foi concluído, a implementação de uma rede *ethernet* é a solução mais viável.

- Relativamente aos motivos de ordem temporal, a *Huf* não definiu uma data para a entrega e posterior execução do sistema de informação, uma vez que este projecto não é prioritário. Motivo pela qual este se encontra agora suspenso, como resultado da crise mundial porque passamos e que está a afectar em especial no sector automóvel.

Tendo presentes as considerações anteriores, a solução que se considera mais vantajosa para o sistema de informação em estudo consiste na utilização de uma rede *ethernet* e no desenvolvimento de uma aplicação à medida com base em linguagens de uso genérico. A escolha recaiu sobre o *Visual Basic 6* para aproveitar as facilidades de integração com OPC, como se poderá verificar no capítulo seguinte.

Antes da decisão sobre a solução a adoptar, para comprovar as conclusões anteriores, foi pedido a uma empresa exterior pela *Huf*, um estudo em que se utilizasse ferramentas de desenvolvimento dedicadas. A empresa em causa utilizava o "*System Platform*" da *Wonderware*. Numa análise muito breve foi possível concluir que, para o sistema em análise, esta solução era mais dispendiosa e mais limitativa de futuras evoluções, embora permitisse uma implementação mais rápida.

A grande vantagem da solução apresentada, mas que não favorecia a *Huf*, residia no facto de se estar perante um software bem testado, onde o desenvolvimento de uma nova aplicação se limita configurar e programar objectos. No entanto, embora a implementação seja mais rápida, a solução é menos flexível uma vez que os objectos estão pré-criados, logo uma alteração na sua estrutura, necessita de uma programação de raiz.

Relativamente aos custos, a análise comparativa revelou:

- Aproximadamente oito vezes mais cara em licenças de software.
- As cartas para redes dedicadas são mais caras assim como os seus cabos e derivadores.

- É necessária mais mão-de-obra para a implementação de várias redes em vês de uma só rede *ethernet*.

3.4 Escolha do Sistema Operativo

O facto de a empresa trabalhar desde o nível de gestão até ao nível do chão da fábrica com a plataforma *Windows*, levou-nos a desenvolver o sistema de informação também recorrendo a esta plataforma. Perante este facto qualquer abordagem passa por utilizar, directa ou indirectamente, as tecnologias COM e DCOM, como base de desenvolvimento para a aplicação que irá gerir a comunicação entre máquinas. Posteriormente será realizado um estudo sobre COM e DCOM.

3.5 Tecnologias de Comunicação

Na linha de montagem escolhida para o projecto, encontramos dois tipos de máquinas relativamente ao tipo de controlo. Será desenvolvida uma abordagem específica na criação do módulo de aquisição de dados para cada tipo de controlo.

Uma das grandes dificuldades encontradas no desenvolvimento desta aplicação prende-se com os *drivers* de comunicação.

Ao analisar as máquinas controladas por PLCs, e tal como já foi referido, deparámo-nos com equipamentos de 5 fabricantes diferentes, o que poderia obrigar ao desenvolvimento de um *driver* de comunicações para os equipamentos de cada fabricante.

Esta situação levou-nos a uma pesquisa para tentar encontrar uma técnica que permitisse desenvolver apenas um *driver* de comunicações.

As máquinas controladas por IPC (Computador Industrial), utilizam uma aplicação que corre sobre a plataforma *Windows* para assim realizar o controlo da mesma. Neste caso a abordagem para a comunicação apresenta-nos dois caminhos distintos, o primeiro é desenvolver uma aplicação que vai correr no computador/controlador da máquina e que será independente da aplicação que controla a máquina. O segundo caminho recorrendo a alteração da aplicação para que nos forneça os dados. Esta possibilidade obriga a uma cooperação com o fabricante o que nem sempre é possível mas é sem duvida a mais segura e rápida e menos dispendiosa.

Em função das conclusões e caminhos definidos anteriormente, foi realizado um estudo para identificar possíveis tecnologias aplicáveis ao projecto.

Foram encontradas duas soluções que estarão na base do sistema de informação e que vão permitir conhecer o desempenho do processo:

1 Solução - Baseada na Tecnologia OPC

2 Solução - Baseada em Objectos Dcom

A tecnologia OPC, será responsável por criar e gerir o interface de dados para as máquinas que são controladas por PLCs. É um standard industrial desenvolvido pela *OPC Foundation* que especifica uma interface de software, que vai permite recolher os dados produzidos pelos dispositivos de automação. No capítulo 5 será analisada a tecnologia OPC em detalhe.

A segunda solução baseia-se em Objectos Dcom, trata-se de uma técnica largamente utilizada nos dias que correm e que se insere na programação por componentes. Esta técnica será aplicada aos equipamentos que são controlados por IPCs, será desenvolvido um cliente e vários servidores que executam a troca de informação via DCOM.

É uma técnica pouco invasiva no que respeita à inclusão de um componente (servidor de dados DCOM) na aplicação que controla a máquina.

O facto da *Huf* manter um contacto estreito com as empresas que desenvolveram as suas máquinas, viabiliza a inclusão de componentes de software nas aplicações das mesmas. Esta estreita ligação vai permitir desenvolver um objecto tipo e pedir aos fabricantes, com os respectivos custos, a inclusão desse software na sua aplicação. Esta técnica será explicada pormenorizadamente no capítulo 4.

3.6 Resultados e Conclusões do Capítulo

No fim deste capítulo passamos a:

- Ter definido a arquitectura geral, a topologia de rede e a ferramenta de desenvolvimento a utilizar no desenvolvimento do módulo.
- Ter definido as tecnologias de comunicação a aplicar no desenvolvimento do módulo de aquisição de dados, em função do tipo de máquina.

Principais conclusões do capítulo:

- A arquitectura a desenvolver na implementação do sistema de informação será constituída por um computador central que troca a informação directamente com as máquinas da linha.
- Será desenvolvida uma aplicação (modulo de aquisição e dados) em *Visual Basic 6*, que utilizará a plataforma *Windows*, e a rede *ethernet* para comunicar com os equipamentos.

- A escolha da ferramenta de desenvolvimento está fortemente influenciada por factores de ordem financeira, volume de transferência de dados, *know-how* dos colaboradores e motivos de ordem temporal.
- O sistema de informação pode registar aproximadamente até 700 registos por minuto. Deve-se por este motivo ter atenção ao desempenho do conjunto PC, aplicação e rede.
- A tecnologia de comunicação a ser empregue para gerir o interface de dados nas máquinas que são controladas por PLCs será o OPC, para equipamentos controlados por IPC será a programação por componentes recorrendo a objectos Dcom.

Fazendo o ponto da situação, para a execução do projecto está definido e justificado:

- Para os dois tipos de controlo identificados na linha será escolhida uma máquina representativa para a realização do projecto.
- Instalar um PC central e uma rede *ethernet* que permitam a troca de informação com os equipamentos da linha.
- Adicionar uma carta de rede *ethernet* ou um conversores *série-ethernet* aos PLCs que realizam o controlo dos equipamentos, para poderem comunicar com a rede.
- Adicionar uma placa de rede aos equipamentos que sejam controlados por IPCs e que não a possuam.
- Desenvolvimento de raiz da aplicação que vai implementar sistema de informação orientado ao processo, utilizando como ferramenta de desenvolvimento a linguagem *Visual Basic 6*
- As tecnologias de comunicação a serem utilizadas na aplicação para a recolha dos dados.(OPC e DCOM)

Nos capítulos seguintes será explicado em que consiste cada uma das tecnologias escolhidas e como foram aplicadas.

Capítulo 4

Tecnologia DCOM

Com já foi referido o módulo de aquisição de dados terá na sua programação duas abordagens diferentes consoante o tipo de controlador que a máquina possuir. Seguidamente será analisada a solução adoptada para as máquinas controladas por IPCs.

Neste capítulo será apresentada a tecnologia Dcom, assim como as principais técnicas associadas. Será apresentado um cliente e servidor que foram desenvolvidos com o objectivo de demonstrar e testar a aplicabilidade desta tecnologia.

4.1 Conceitos e Fundamentos de COM e DCOM

Neste ponto, serão descritos os tópicos mais relevantes do estudo desenvolvido para compreender os principais conceitos e fundamentos de COM e DCOM. Serão apresentados conceitos que apresentam o caminho seguido, justificando as decisões tomadas.

O desenvolvimento de aplicações em sistemas operativos da *Microsoft* leva-nos à utilização do COM e DCOM, mesmo que por vezes não nos darmos conta da sua utilização.

Os tópicos seguintes tiveram com referência essencialmente, os documentos presentes no sitio da *Microsoft-MSDN* [Com09], [Dcn09] e [Dcn09b], os documentos de Miguel Monteiro [Mig09] na disciplina de Tecnologias de Distribuição e Integração e as apresentações da disciplina de Sistemas Distribuídos de Maria Toledo [Tol09]

4.1.1 História

“A especificação COM (Component Object Model) surgiu em 1993, na sequência da evolução de tecnologias que permitiam a integração de aplicações e a criação de documentos compostos nos sistemas operativos Windows. Essas tecnologias mais antigas ficaram conhecidas por DDE

(Dynamic Data Exchange), que definiu protocos e serviços de comunicação entre aplicações para a troca de dados, e OLE (Object Linking and Embedding) permitindo a criação de documentos por várias aplicações, activando a responsável pela criação e edição de cada parte. O COM surgiu como uma forma de partilha, a nível binário, de código previamente desenvolvido e compilado, usando um paradigma orientado a objectos. Sendo uma especificação de formato binário, permite uma independência das linguagens de programação, desde que estas sejam capazes de gerar esse formato binário.

Paralelamente, e desde os anos 80, a tecnologia da invocação remota de procedimentos tinha vindo a evoluir, culminando com uma especificação da Open Sofware Foundation (OSF) designada por DCE (Distributed Computing Environment). Esta especificação, muitas vezes designada simplesmente por RPC (Remote Procedure Call), permite a comunicação entre aplicações a executar em computadores diferentes através da invocação remota de rotinas, com a passagem de parâmetros e resultados. A OSF/DCE RPC foi implementada já no Windows NT 3.1.

Combinando RPC com COM foi possível activar e invocar objectos COM a executar em processos diferentes dos clientes (quer na mesma máquina, quer em máquinas diferentes), dando origem à especificação DCOM (Distributed COM), formalizada em 1996." [Mig09, Cap. História].

Também no ano de 1996 são desenvolvidos os objectos *ActiveX*, que trabalham sobre COM e DCOM. Um ano mais tarde em 1997, é introduzido o MTS (*Microsoft Transaction Server*) que melhora a gestão de transacções de componentes, recursos e controlo de segurança no COM e DCOM, dando lugar ao COM+, que assim simplifica a criação de aplicações escaláveis.

A tendência nos dias que correm, segue os caminhos do .NET e o protocolo *SOAP*, no entanto é possível continuar a utilizar COM e .Net como o referenciado em artigos no site da Microsoft.

4.1.2 Introdução

A especificação COM é um modelo criado para desenvolver componentes orientados a objectos, independentemente da plataforma. A biblioteca COM fornece um conjunto de serviços (*APIs*), que são parte integrante do Windows e que podem estar disponíveis em outros sistemas operativos através da instalação de um pacote.

A extensão do COM para ambientes distribuídos é denominada de DCOM (*Distributed Component Object Model*), gere os protocolos de rede de baixo nível, e oferece autentificação e encriptação de dados.

Os componentes podem estar no mesmo processo, em processos diferentes ou máquinas diferentes, quando temos um componente no mesmo processo ou em processos diferentes utilizamos a especificação COM para a troca de informação, nas figuras 4.1 e 4.2 temos respectivamente a ilustração simbólica de um componente COM num mesmo processo e

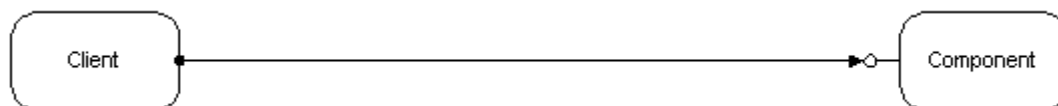



Figura 4.1 – Componente COM num mesmo processo, Adaptado de [Dcn09]

a ilustração de como as bibliotecas de run-time COM/DCOM implementam a comunicação entre o cliente e o componente em processos diferentes.

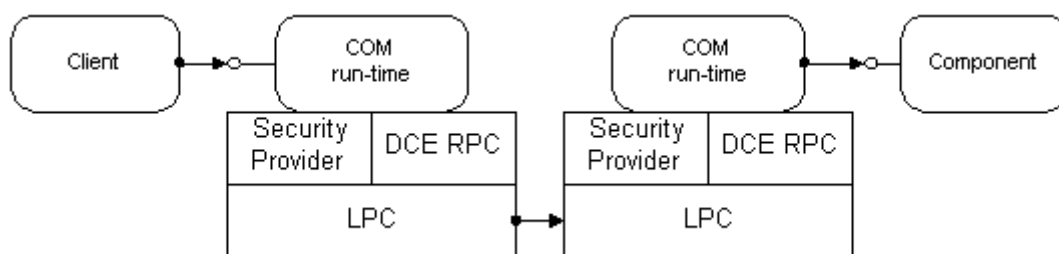



Figura 4.2 - Componente COM em diferentes processos, Adaptado de [Dcn09]

Quando temos processos que pretendem trocar informação em diferentes máquinas recorremos à especificação DCOM, na figura seguinte pode-se ver como se implementa a arquitectura DCOM.

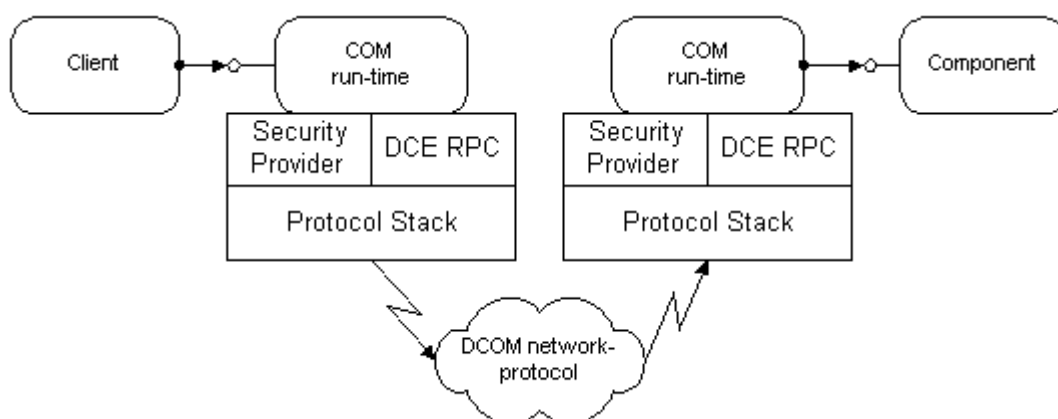



Figura 4.3 - Componente DCOM em diferentes máquinas, Adaptado de [Dcn09]

4.1.3 Interfaces

Os serviços implementados pelos objectos COM, são expostos através de interfaces que representam o único ponto de contacto entre os clientes e os objectos. Os interfaces são imutáveis porque temos de criar uma nova interface cada vez que queremos adicionar uma nova funcionalidade, mesmo que seja uma extensão de uma interface antiga.

Todas as interfaces derivam directa ou indirectamente, da interface *IUnknown*, uma vez que esta é a responsável por criar e realizar a manutenção das interfaces, apresenta três métodos o *QueryInterface* (obtém ponteiro para uma interface), *AddRef* (incrementa o contador)e o *Relase* (decremento do contador).

Cada objecto têm um CLSID (*class Identifier*) e cada interface tem um IID (*Interface Identifier*, estes são identificados por um GUID (*Global Unique Identifier*) que fica armazenado no registo do *Windows*. Os identificadores podem ser gerados por ferramentas *GUIDGEN* ou recorrendo a função *CoCreateGuid*.

O DCOM implementa mecanismos de reutilização como seja a Contenção/Delegação e Agregação.

4.1.4 Arquitectura Cliente e Servidor

Em COM e DCOM a arquitectura cliente-servidor, é baseada no seguinte principio:

- O cliente é uma aplicação que faz uso de um objecto Dcom, recorrendo para isso aos interfaces do objecto.
- O servidor é um componente/objecto que implementa as classes DCOM e respectivos interfaces.

Podemos encontrar três tipos de servidores: (i) *In-process* (DLL) cliente e servidor no mesmo processo; (ii)*Local Object Proxy* (EXE) cliente acede ao servidor em processos diferentes, mas na mesma máquina; (iii) *Remote Object Proxy* (EXE), cliente pode aceder ao servidor em máquinas diferentes. DCE RPC => DCOM.

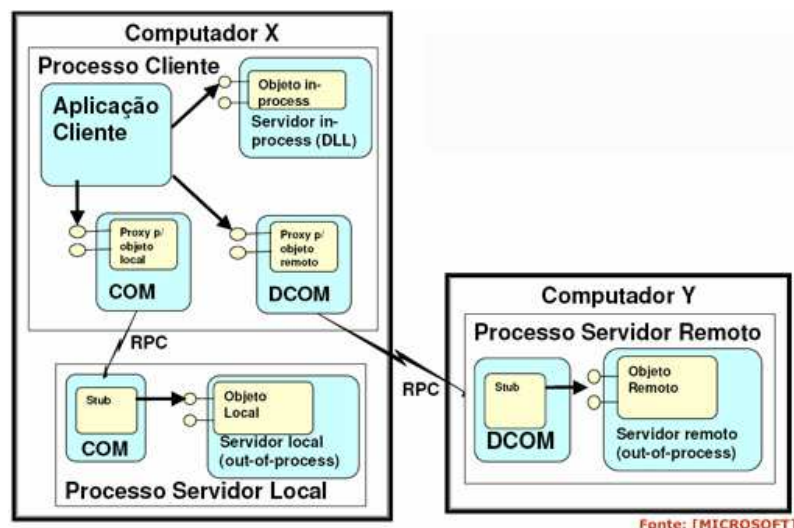



Figura 4.4 – As três formas de um cliente trocar informação com o servidor

.

4.2 Aplicação Cliente e Servidor em DCOM

Passemos a descrever o caminho seguido no desenvolvimento de uma aplicação para teste de um cliente e servidor, que permita recolher os dados das máquinas que são controladas por computadores industriais (IPCs).

O objectivo desta aplicação é permitir utilizar uma técnica baseada na especificação DCOM, que permita sem uma grande intervenção na aplicação que controla a máquina, recolher informação do seu ciclo de trabalho.

Aplicando os conceitos da especificação DCOM, vamos desenvolver um servidor que implemente uma classe DCOM, e os respectivos interfaces. A classe DCOM recolhe a informação dos estados da máquina e através dos interfaces da mesma, o cliente acede a informação no servidor e regista essa informação na base de dados do sistema de informação.

O servidor será do tipo *Remote Object Proxy*, para permitir que o cliente possa comunicar com ele sem que esteja instalado na mesma máquina. Este pode estar ou não incluído na aplicação que controla a máquina, temos assim duas possibilidades de implementarmos as trocas de informação entre o servidor e a aplicação.

O Servidor não tem interface para o utilizador, tem de ser registado no sistema operativo da máquina, passando assim a correr na maquina sempre que solicitado por uma aplicação cliente na própria máquina ou em qualquer máquina da rede.

O cliente desenvolvido tem o seguinte interface, que permite: (i) iniciar ou parar a recolha, (ii) registar os dados na base de dados, (iii) visualizar o último registo, (iv) fazer um teste ao servidor para verificar se temos comunicação, ou se o mesmo está operacional.

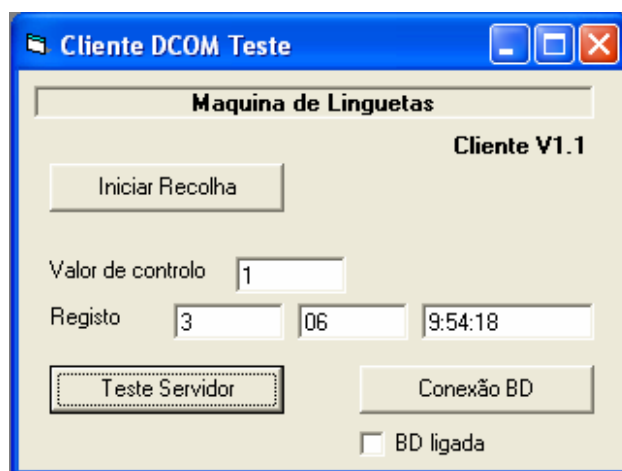



Figura 4.1 – Interface do Cliente DCOM

O teste realizado à aplicação cumpriu as funções desejadas, surgiram alguns problemas na fase de instalação e registo do servidor, mas foram ultrapassadas recorrendo as ferramentas que o VB6 tem para esta situação.

Questões de desempenho não se põem, uma vez que o DCOM é uma ferramenta por excelência para aplicações do género, e portanto capaz de garantir a troca de informação com rapidez e qualidade.

4.3 Resultados e Conclusões

Os testes realizados com o cliente e servidor desenvolvidos permitem demonstrar que a solução encontrada baseada em componentes/objectos DCOM, a aplicar às máquinas que são controladas por IPCs, cumpre de forma satisfatória o objectivo da recolha dos dados.

A maior dificuldade no desenvolvimento desta aplicação, aparece na configuração e registo dos componentes nas respectivas máquinas. Esta dificuldade já característica da tecnologia DCOM, foi ultrapassada com uma metodologia rígida de trabalho, no entanto no caso do VB6 pode-se utilizar uma técnica que desenvolve este trabalho de uma maneira semi-automática, trata-se pois de criar um *Package* para o cliente e servidor. O VB6 ao criar um *Package*, vai mecanizar o processo e cria automaticamente todos os ficheiros que são necessários ao registo dos componentes, cria também um executável que será o responsável pela instalação e registo dos respectivos componentes. Depois é apenas correr o executável nas máquinas que se desejar.

Não foram realizados testes na máquina de linguetas porque como já foi referido a empresa suspendeu o desenvolvimento deste projecto, devido a crise que se faz passar no sector automóvel. Perante esta situação não foi possível pedir ao fabricante que incluísse um componente/objecto no seu software que nos devolvesse os instantes temporais da entrada dos estados da máquina.

Capítulo 5

Apresentação da Tecnologia OPC

Neste capítulo é apresentado um estudo sobre a tecnologia OPC, com especial incidência na especificação escolhida: *OPC Data Access*.

5.1 Estudo Sobre a Tecnologia OPC

Seguidamente vão ser expostos os tópicos mais relevantes do estudo, desenvolvido para entender os conceitos e técnicas em que a tecnologia OPC se baseia.

Os tópicos seguintes dos sub capítulos 5.1 e 5.2, tiveram com referência essencialmente os documentos, presentes no sítio da *OPC Foundation* [OPC08b] e [OPC08c], de Paulo Portugal [Pau05] e [Pau06], de H. Kirrmann [Kir08a] e [Kir08a], e de Adriano Puda [Adr08], e ainda a referência da *OPC Foundation* [Iwa06].

5.1.1 Conceito

No mercado existem inúmeros sistemas para ajudar na gestão e controlo das áreas de produção das empresas, como sejam os, SCADAS, os HMIs, os MESs, os LIMSs, os ERPs, os *Loggers* de alarmes e Eventos (Registadores), *Logs* de Históricos, Etc.

Muitos desses sistemas tendem a ser aplicações isoladas umas das outras ou pelo menos com pouca integração entre elas. Os seus protocolos de comunicação proprietários, tornam os sistemas de análise desconexos dentro da empresa. Perante a necessidade de evolução ou expansão de uma qualquer empresa, o engenheiro naturalmente escolhe o caminho mais fácil e acaba integrando ferramentas da mesma família/marca, mesmo que no mercado existam soluções mais interessantes para as suas necessidades. Esta atitude garante a total compatibilidade entre os sistemas na troca de informação.

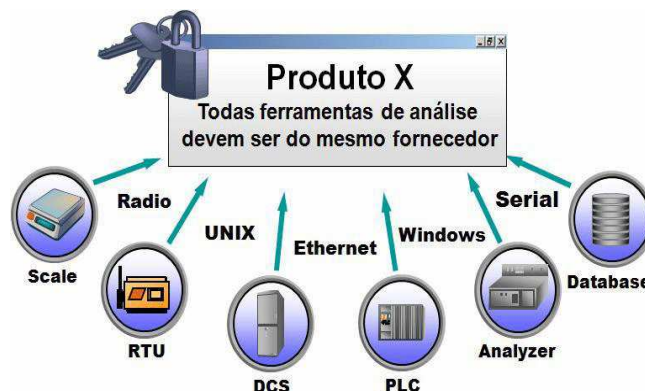



Figura 5.1 - Falta de conectividade dos sistemas tradicionais, Adaptado de [Adr08]

Se pensarmos ao nível da gestão, as aplicações que preenchem as necessidades específicas deste nível (contabilidade, gestão de custos, planeamento da produção, etc), tendem a apresentar uma integração maior. Mas vão surgindo lacunas que vão dificultar a troca de informação.

A tecnologia OPC (*OLE for Process Control*) vai permitir a integração dos dados da empresa, sejam estes provenientes de qualquer dos níveis da mesma.

Figura 5.2 - Camada de comunicação standard OPC, Adaptado de [Adr08]

5.1.2 Standard OPC

A tecnologia OPC começou por ser denominada "*OLE for Process Control*" mais tarde passou a chamar-se "*Open Process Control*", é desenvolvida pela "*OPC Foundation*" e trata-se de um standard industrial aberto. O OPC separa os sistemas das dificuldades de comunicação ao criar uma camada única standard que vai permitir a fácil integração dos diversos sistemas da empresa. Pode abranger desde o mais simples instrumento na linha de montagem até aos sistemas de gestão como os ERPs.

Sendo um standard de comunicação aberto, apresenta como principal objectivo permitir a interoperabilidade vertical entre os sistemas que se encontram dentro de uma organização.

Em 1995, a reunião de um conjunto de fabricantes de sistemas de automação industrial cria a *OPC Foundation*. que em 1996 apresenta a primeira versão funcional do OPC. Nos dias que correm fazem já parte da organização mais de 300 empresas, sendo esta que define as especificações do OPC, e que procura melhorar e evoluir a tecnologia.

A sigla OPC que traduzida para português significa Controlo Aberto de Processos, denomina uma tecnologia que está baseada nas tecnologias da Microsoft como sejam o OLE (*Object Linking and Embedding*), COM e DCOM. Trata-se de um conjunto de interfaces, métodos, objectos e propriedades de comunicação que foram reunidos numa especificação standard e aberta para acesso público. Perante esta característica o programador para desenvolver uma aplicação terá de consultar as especificações que a *OPC Foundaction* desenvolveu e criar o seu próprio interface, que responda as suas necessidades.

O OPC é disponibilizado através de uma biblioteca de software (DLL) que pode ser invocada por aplicações clientes, escritas em *Visual Basic*, C++, C# ou qual outra linguagem da família da Microsoft. A DLL têm dois tipos de interfaces, o *Automation Interface* (*Visual Basic*) e *Custom Interface* (C++).

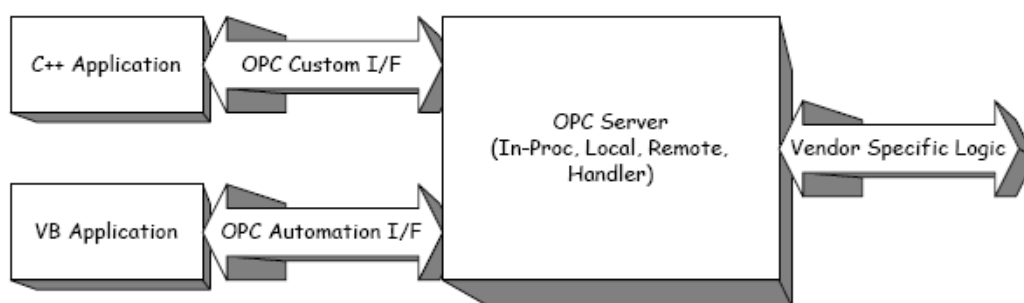



Figura 5.3 - Tipos de interfaces para OPC, adaptado de [OPC08b]

5.1.3 Potencialidades e Vantagens do OPC

Um exemplo concreto das potencialidades do OPC, é o verificado em qualquer software de automação, pois cada aplicação dentro desta área tem os seus próprios *drivers* de comunicação para cada marca de PLCs. Ou seja o *driver* da empresa de software x não funciona com a aplicação da empresa de software y para o mesmo PLC. A figura seguinte exemplifica a ideia anterior, que é de termos de desenvolver um *driver* para cada marca de equipamento:

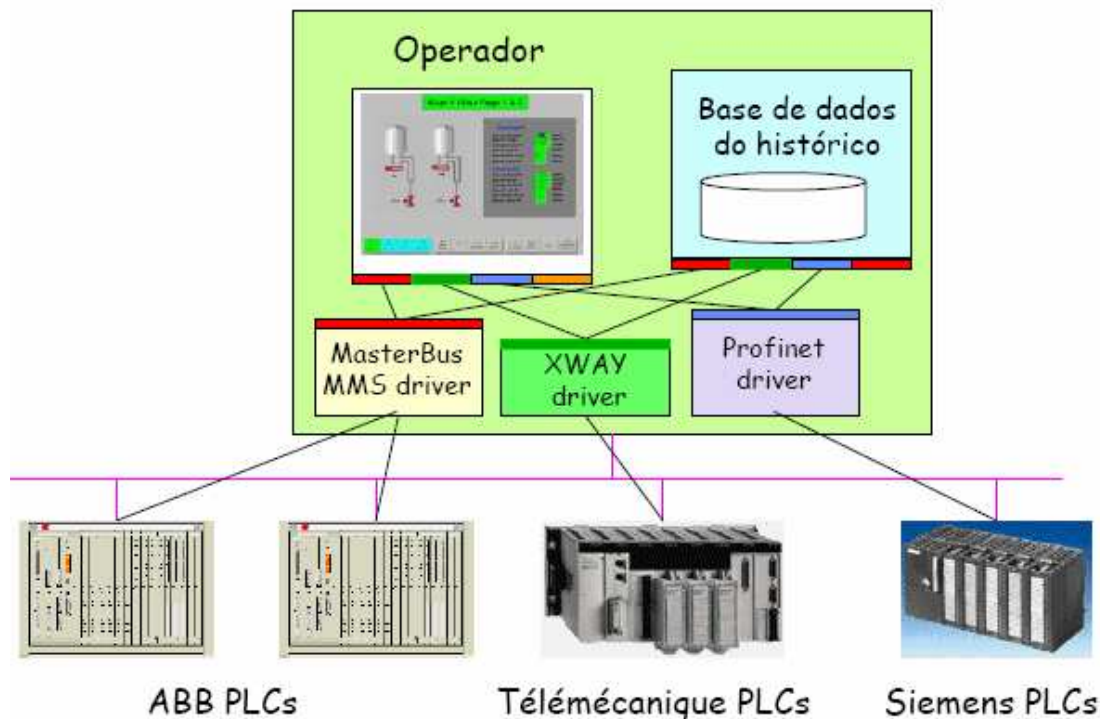



Figura 5.4 – Antes do OPC, adaptado de [Pau05]

É este o principal motivo que leva a que se tenha muito cuidado na aquisição de software e hardware, para que estes possam comunicar entre si e sem conflitos. Esta situação leva a que na fase de projecto se opte por soluções que sejam desenvolvidas pelo mesmo fabricante e portanto perfeitamente compatíveis. Este motivo leva também por vezes, empresas que pretendem evoluir o seu processo de fabrico a escolher um produto que não seja o mais atractivo, mas que garanta a compatibilidade com o que já esta implementado.

Ao basear o OPC na tecnologia OLE, que sendo uma tecnologia *Windows*, visa entre outras coisas garantir que software e hardware possam trocar informação sem que o software tenha desenvolvido um *driver* específico para o hardware.

Um exemplo bem conhecido para explicar a ideia anterior é o suporte de impressão incorporado no sistema operativo *Windows*, que obriga a que todas as aplicações que pretendam trabalhar com os serviços de impressão, tenham que possuir somente um *driver* para esse suporte de impressão e não um para cada impressora existente no mercado.

Com o OPC estendemos o conceito/benefício da tecnologia OLE à área industrial, a partir de aqui as empresas de automação desenvolvem servidores OPC para os seus equipamentos e os programadores desenvolvem aplicações (clientes OPC), e que vão através do servidor OPC trocar informação. Desaparecendo assim a necessidade de desenvolver inúmeros *drivers* de comunicação.

Fazendo aqui um aparte ao projecto, verifica-se que o principal objectivo para a escolha do OPC para uma parte do projecto se concentra na principal vantagem da tecnologia OPC.

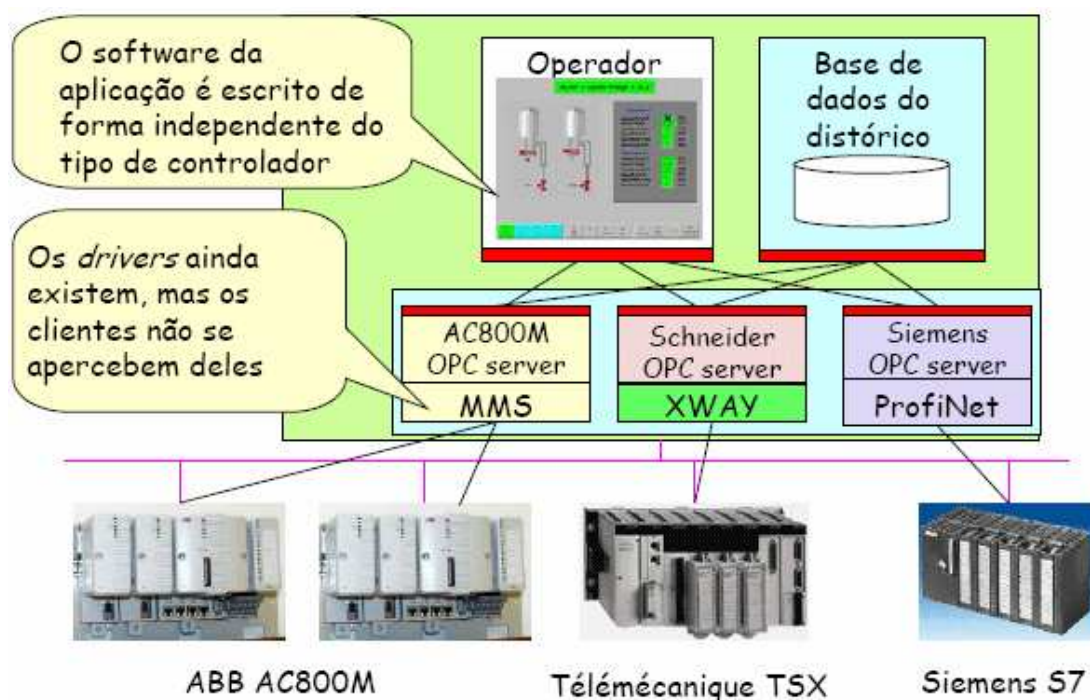



Figura 5.5 – Com OPC, adaptado de [Pau05]

O software de controlo (cliente OPC) a ser desenvolvido não vai ter em conta qual o tipo de controlador, mas apenas o standard OPC, desta forma os *drivers* existem mas não são vistos por parte dos clientes (figura 5.5).

Vamos perante esta técnica ter como principais vantagens:

- Desenvolver apenas um *driver* de comunicação, reduzindo assim o esforço de desenvolvimento.
- Reduzir ou acabar com incompatibilidades e conflitos de acesso.
- Estabelecer um standard.
- Aumentar a Integração entre os distintos fabricantes.
- Menor dependência de Hardware e Software.
- Aumentar a flexibilidade.

Em concreto o OPC vai permitir aos fabricantes reduzir os seus custos com conectividade e aplicar as suas competências no desenvolvimento dos seus produtos, por outro lado os clientes ganham uma maior flexibilidade nos equipamentos, e passam a escolher o software em função das suas potencialidades e não se preocupam se o mesmo tem ou não o *driver* para o seu equipamento.

5.1.4 A Arquitectura Cliente-servidor do OPC

O princípio de funcionamento do OPC é baseado na tão usada arquitectura cliente-servidor, ou seja os servidores OPC fornecem dados aos clientes OPC, em que o servidor têm a função de recolher os dados do equipamento.

Figura 5.6 - Arquitectura cliente-servidor do OPC

Um aspecto importante a salientar é o facto de um servidor poder fornecer dados para mais do que um cliente, e que um cliente pode receber dados de mais do que um servidor, e que também vários servidores podem ser executados em paralelo. Estas possibilidades de configuração permitem uma enorme flexibilidade e variedade de configurações para a recolha de dados num sistema, isto é, desde que os elementos do sistema sejam compatíveis com OPC.

É necessário ter em conta que o OPC não elimina o protocolo proprietário dos equipamentos ou aplicações, mas serve de tradutor quando converte as instruções do protocolo OPC em instruções do protocolo proprietário ou vice-versa.

É então necessário desenvolver um servidor específico para cada um dos protocolos de comunicação existentes. Tarefa que cabe normalmente as empresas que produzem os equipamentos. Existem no mercado empresas que se especializaram no desenvolvimento de servidores e soluções para a grande maioria dos sistemas de automação conhecidos, como sejam a *KEPware* e a *Matrikon.*

Segundo um estudo pela *OPC foundation* o OPC é a tecnologia preferida para assegurar a conectividade de 78% dos MÊS (*Manufacturing Execution Systems*), 75% dos SCADAs, 68% dos PLCs e 53% dos ERP (*Enterprise Resource Planning*).

5.1.5 Tipos de Especificações

Perante o principal objectivo de solucionar o problema dos *drivers* de comunicações proprietárias, a "*OPC Foundation*" primeiramente criou um modelo standard num mercado onde só existiam soluções customizadas / dedicadas, que se viria a chamar **OPC DA (Data Access)**, que permite a troca de dados em tempo real.

Com esta especificação temos a possibilidade de saber e alterar o estado de qualquer variável do processo a controlar. Os clientes típicos desta especificação são as aplicações de visualização do processo e o controlo supervisionado.

As variáveis do processo representam o evoluir do mesmo, e podem ser enviadas para o cliente de três maneiras distintas, a pedido, quando o seu valor muda ou perante um determinado intervalo de tempo.

Perante os bons resultados que o OPC apresentava, começam a surgir novas especificações tais como:

OPC HDA (Historical Data Access)

Esta especificação apareceu para podermos ter acesso a dados em históricos, recorrendo ao OPC. Permite também aceder aos dados que possam estar armazenados num equipamento e transferir esses dados entre aplicações, tais como o software que controla o Histórico e um SCADA, ou mesmo criar uma tabela em *Excel* com esses dados.

O OPC HDA permite a utilização do standard OPC quando é necessário analisar dados provenientes de diversos equipamentos e ou sistemas diferentes, para posteriormente trabalha-los numa aplicação cliente única.

Os Clientes principais do OPC HDA são gráficos de tendências e históricos.

OPC AE (Alarms & Events)

A especificação OPC AE, apareceu para ocupar um lugar muito especifico de permitir a transmissão de alarmes e eventos e toda a mecânica a eles anexa, como sejam a sua classificação, intervenções do operador, parâmetros de severidade do alarme, área de ocorrência, …

Permite também criar bases de dados de históricos ou de informações sobre a operação dos principais eventos dos processos.

Os seus principais clientes são os *Loggers* de alarmes e Eventos

OPC UA (Unified Arquitectura)

Sendo a última especificação a ser desenvolvida, está numa fase de finalização da especificação. Começou a ser desenvolvida em 2004, e pretende extravasar o OPC para alem do ambiente industrial. Vai como o próprio nome sugere, unificar as especificações OPC,

principalmente a OPC DA, OPC HDA e OPC AE. Esta unificação vai permitir que um equipamento ao ser compatível com a especificação OPC UA, possa trocar informação com outro equipamento que possua uma das especificações OPC.

Esta evolução tem em linha de conta as tendências que a Internet está a seguir, pois torna-se multi-plataforma e recorre a tecnologias orientadas para a *web* como sejam TCP/IP, HTTP, XML e SOAP, deixando o COM e DCOM para segundo plano. Esta mudança vai permitir utilizar o OPC em Plataformas *Windows* e *Linux* e funcionar também em equipamentos compactos como PDAs

A especificação OPC UA está a ser criada por partes, apresentando algumas já finalizadas, e deve estar completa dentro de muito pouco tempo, mas reivindica já que quer ser o padrão dominante na próxima década.

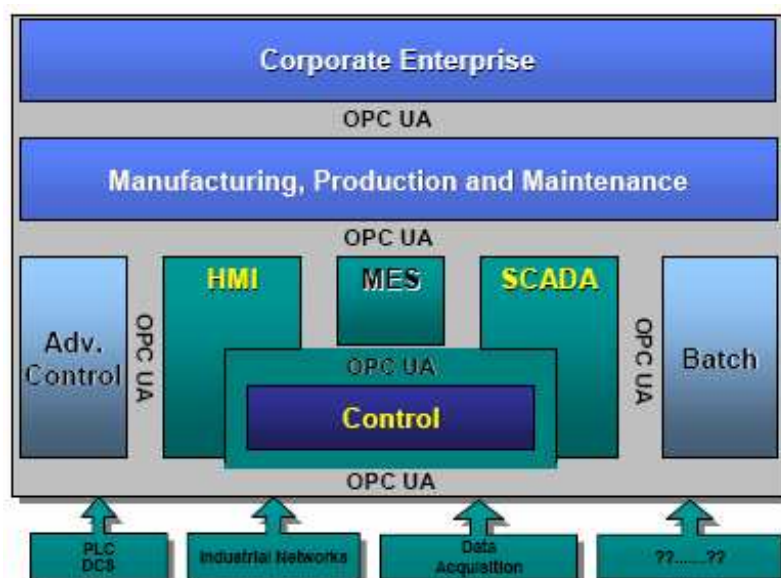



Figura 5.7 - OPC UA: Integração vertical de todos os sistemas da empresa, Adaptado de [OPC09e]

A especificação OPC DA é largamente utilizada nos dias que correm, no entanto as especificações OPC HDA, OPC AE, são muito pouco utilizadas e em áreas muito especificas. Á especificação OPC UA começa a aparecer, mas os fabricantes não apresentam ainda soluções para ela.

5.2 Justificação da Especificação Seleccionada

Depois de analisar as especificações existentes, em princípio, poderiam ser utilizadas no desenvolvimento da aplicação em estudo tanto a especificação OPC DA como a especificação OPC AE.

No entanto, a utilização da especificação OPC AE foi afastada logo no início do projecto, uma vez que algums dos fabricantes de autómatos programáveis não apresentam servidores OPC AE pois, tal como já foi referido, trata-se de uma especificação pouco usual e muito específica.

A especificação OPC DA é suportada por todos os fabricantes com representação significativa no mercado de equipamentos para a automação industrial. e em especial no mercado de PLCs. Analisando as marcas que estavam presentes na linha, todas elas forneciam um servidor OPC DA, desenvolvido pelo próprio fabricante. No entanto se fosse confirmado que uma das marcas não tinha o seu próprio software, restava-nos procurar numa das empresas que se dedicam ao desenvolvimento de servidores OPC, de que são exemplo a *Kepware* e a *Matrikom*. Estas empresas desenvolvem servidores OPC DA, permitindo a comunicação com equipamentos que possuam ou não servidor deste tipo, Por vezes estas empresas apresentam um desempenho do seu servidor superior ao da marca original.

Depois de escolhida a especificação a ser utilizada foi consultado o manual disponibilizado pela *OPC Foundation* [OPC08c] e exemplos de aplicações básicos para começar a desenvolver a primeira aplicação. Esta teve como objectivo principal implementar um cliente OPC capaz de trocar informação com um qualquer servidor OPC DA.

De seguida, será apresentada uma breve descrição da especificação que vai permitir compreender melhor o funcionamento desta, e concluir a razão da sua escolha.

5.2.1 OPC Data Access

A especificação OPC DA como já foi referido, destina-se a recolher ou alterar o estado de qualquer variável do processo a controlar, esta foi desenhada para aceder a dados de um servidor em rede.

O desenvolvimento de um sistema que implemente a especificação OPC DA, começa pela configuração do servidor OPC DA. Nesta fase, são escolhidas as posições de memória utilizadas pelos dispositivos para o controlo do processo, e de seguida essas posições de memória são definidas como *TAGS* ou *ITENS* no servidor. Podem-se utilizar os dois termos (*TAGS* ou *ITENS*), vamos adoptar a segunda denominação por está se aproximar mais às indicações da especificação.

A partir deste momento todas as variáveis relevantes para o processo passam a estar mapeadas no servidor OPC, e portanto disponíveis para uma qualquer operação.

Centrando-se agora no lado do cliente (aplicação a ser desenvolvida e que vai aceder ao servidor), e analisando a estrutura tipo para uma aplicação cliente, esta cria grupos de variáveis denominadas de Itens e organiza-os independentemente do servidor. De seguida vai registar a

sua estrutura no servidor criando assim os objectos que permitem a troca de informação no processo.

A título de exemplo, e para entender de forma mais esclarecedora a ideia apresentada no paragrafo anterior, na imagem seguinte pode-se visualizar uma arquitectura tipo, onde pode ser interpretado a estrutura implementada pelo cliente e servidor utilizando a especificação OPC DA.

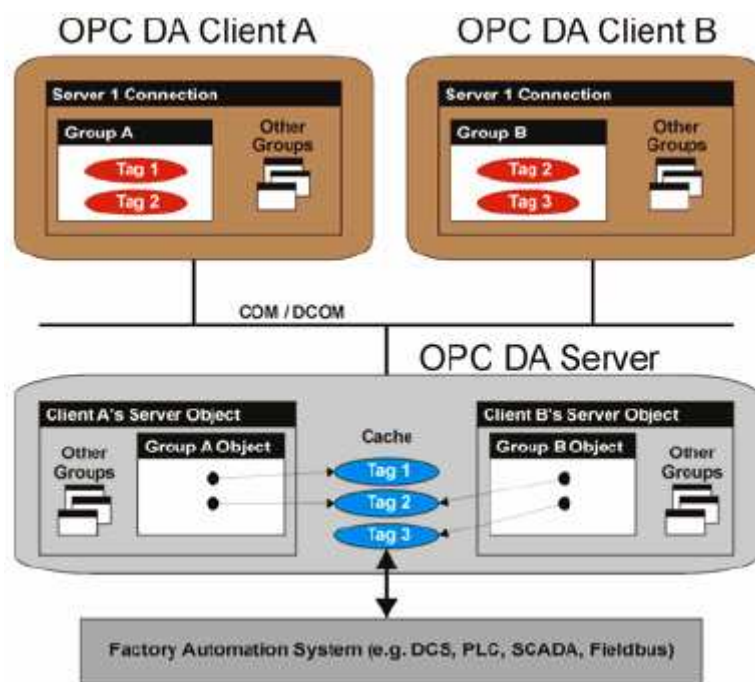



Figura 5.8 – Arquitectura tipo OPC DA, adaptado de [Adr08]

Para criar uma aplicação capaz de implementar a lógica visualizada na arquitectura da figura anterior, existem dois tipos de interfaces, o *OPC Custom* e o *OPC Automation*. O interface que vamos analisar será o *OPC Automation*, uma vez que é o utilizado no projecto. A escolha recaiu naturalmente sobre o *OPC Automation*, pois este é o único a funcionar com as ferramentas de programação da *Microsoft* e foi inicialmente desenvolvimento para trabalhar precisamente com *Visual Basic 6*.

Em termos de programação temos que, a instância do servidor é vista por parte do cliente da seguinte forma:

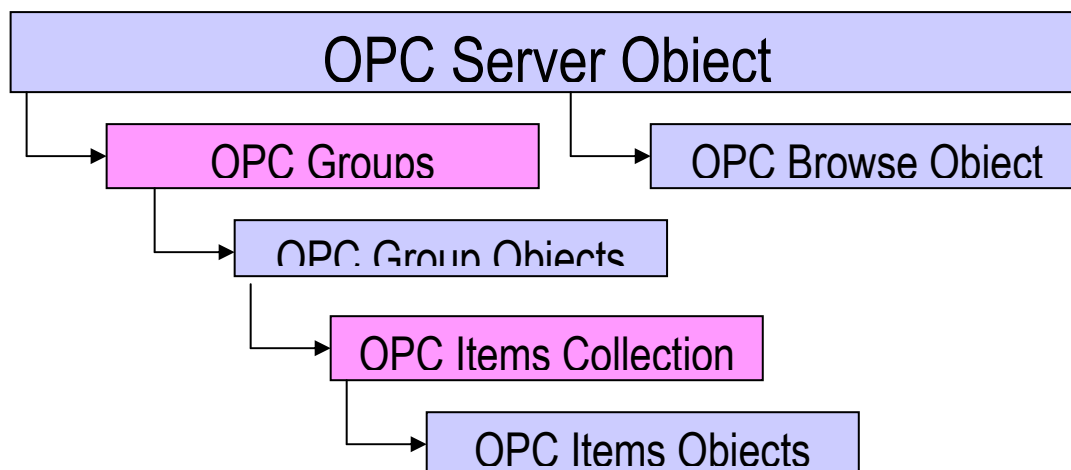



Figura 5.9 – Hierarquia de objectos da OPC Automation, adaptado de [OPC08c]

Analisado a imagem anterior conclui-se que esta hierarquia de objectos leva o cliente a criar vários objectos que estão encapsulados uns nos outros para aceder a um item.
Assim, como exemplo, temos que, ao criar uma instância no servidor, vamos invocar o objecto *OPCServer* e este objecto por sua vez cria o objecto *OPCGroups* e *OPCBrowser*.

Explicando a função principal de cada objecto conclui-se que:

- O objecto ***OPCServer*** cria e gere a conexão entre a aplicação e o servidor.
- O objecto ***OPCBrowser*** é criado pelo objecto *OPCServer* e é responsável por permitir a navegação pelo servidor.
- O objecto ***OPCGroups*** é criado pelo objecto *OPCServer* e representa a colecção dos grupos criados pelo cliente.
- O objecto ***OPCGroup*** mantém informação do estado dos mecanismos de acesso aos itens.
- O objecto ***OPCItems*** representa a colecção dos itens de um grupo
- O objecto ***OPCItem*** representa o objecto que o cliente criou na estrutura e mantém a informação relativa ao objecto.

Cada um destes objectos têm propriedades, membros e eventos que vão permitir o controlo da informação de modo a implementar os objectivos previstos para a aplicação.

Cada Item é representado por três propriedades dinâmicas:
1º **Valor** – indica o conteúdo da variável e é formatado por tipos, no caso de operações de escrita é o único utilizado.
2º ***Time-stamp*** – instante de tempo em que o item foi transmitido do dispositivo de automação para o servidor.

3º **Qualidade** – indica a validade da leitura

Existem várias formas de um cliente trocar dados com um servidor OPC DA, relativamente à leitura de dados podemos ter:

Leitura síncrona – Trata-se de um método de leitura que ocupa o cliente OPC enquanto os dados solicitados não são entregues pelo dispositivo. É de fácil implementação mas não é a mais eficiente para o sistema.

Leitura assíncrona – Permite que o cliente OPC continue a trabalhar enquanto o servidor recolhe os dados pretendidos. Para realizar esta função é utilizada uma função de *callback*. O servidor OPC está preparado para aceitar vários pedidos de dados de vários clientes OPC, e agrupar todas essas solicitações da melhor forma possível para enviar posteriormente aos dispositivos. Se for aplicável é recomendado o seu uso pois trata-se do tipo de leitura mais eficiente.

Leitura *refresh* – O cliente OPC vai ler todos os dados do processo disponíveis no servidor, mesmo os que não estão activos.

Leitura por subscrição – O servidor OPC lê os dados do processo periodicamente com uma taxa de actualização que é estabelecida pelo cliente OPC. Os valores são enviados ao cliente OPC sempre que se verifique uma mudança num item. Existe ainda a possibilidade de ser definido uma *Deadband* que ao ser excedida transmite o valor ao cliente, a função de *Deadband* aplica-se apenas a valores analógicos.

Relativamente à escrita, podemos ter dois tipos:

Escrita síncrona – Normalmente é realizada directamente nos dispositivo mas na última versão da especificação é possível também registar na *cache*. No entanto também ocupa o cliente enquanto a confirmação da conclusão do comando não é devolvida ao cliente.

Escrita assíncrona – Permite que o cliente OPC continue a trabalhar enquanto o servidor não recebe a confirmação da escrita do valor.

Não esquecer que por questões de segurança, na escrita de um valor é sempre devolvido a indicação de sucesso ou falha do comando executado.

A partir da última versão da especificação OPC DA é possível que uma leitura ou escrita seja feita directamente para o dispositivo ou para a *cache* e, posteriormente, para o dispositivo, e vice versa. A utilização da *cache* torna o processo mais estável, mas pode tornar o processo mais lento quando a largura de banda disponível é limitada.

Na figura seguinte podemos ver esquematicamente como é feito o acesso a uma variável:

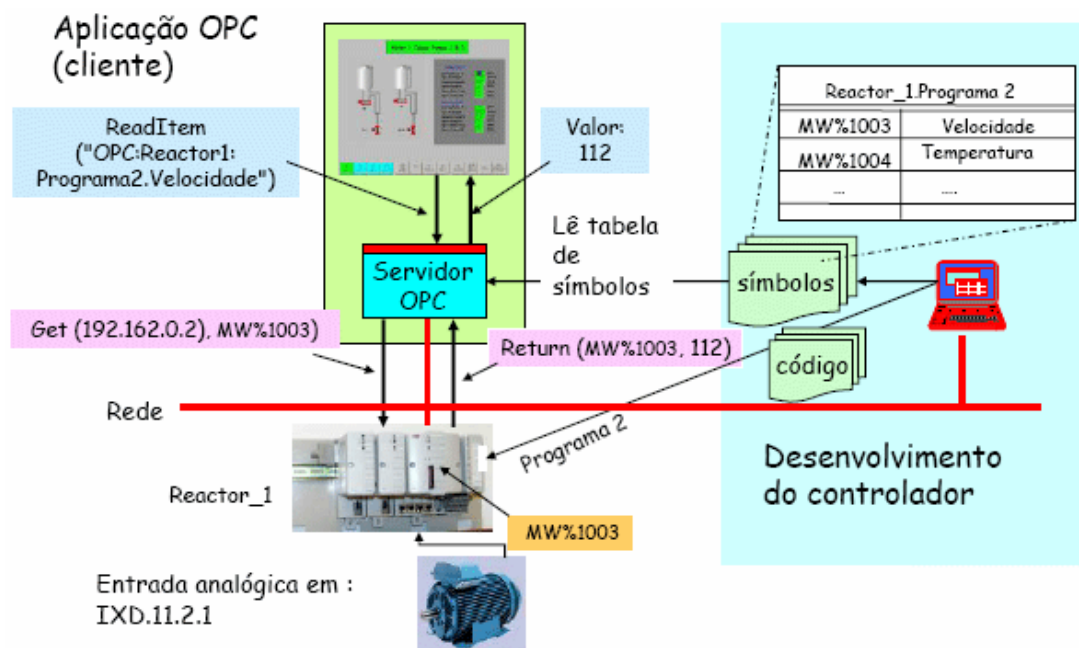



Figura 5.10 – Acesso a uma variável em OPC DA, adaptado de [PAU06]

O OPC DA funciona de acordo com o paradigma da memória partilhada, uma vez que um novo valor de um item é escrito por cima do valor anterior. Esta característica leva a que o servidor não garanta que clientes diferentes tenham a mesma imagem do processo. Cabe ao programador implementar ferramentas capazes de garantir a recolha de toda a informação.

5.3 Resultados e Conclusões

No fim deste capítulo passamos a:

- Conhecer os conceitos e fundamentos da Tecnologia OPC.
- Conhecer a especificação OPC DA.

Principais conclusões do capítulo:

- Vamos aproveitar a principal vantagem da tecnologia OPC, que é separar os sistemas das dificuldades de comunicação, ao criar uma camada única standard e permitir assim a fácil integração dos diversos sistemas da empresa, para desenvolver a parte do módulo de aquisição de dados que vai operar com as máquinas controladas por PLCs.

- A utilização da tecnologia OPC e em concreto da especificação OPC DA vai permitir desenvolver apenas um *driver* de comunicação e reduzir o esforço de desenvolvimento.
- A especificação OPC DA, permite implementar um modelo standard que realiza a troca de dados em tempo real.

Capítulo 6

Avaliação do Desempenho OPC

Neste capítulo, será inicialmente apresentado um Cliente OPC DA que foi desenvolvido para efectuar um primeiro contacto e teste da tecnologia OPC. Posteriormente, foi adicionado a este cliente a possibilidade de registo dos dados recolhidos numa base de dados *Access*.

As aplicações desenvolvidas, proporcionaram um conhecimento pormenorizado de como a tecnologia OPC e, em particular, a especificação DA funcionam, e também como são aplicadas "no terreno". Vão permitir adquirir boas práticas no desenvolvimento de produtos da família OPC, assim como conhecer as capacidades da tecnologia. Obtêm-se nestes moldes um rendimento superior no desenvolvimento futuro de software.

6.1 Desenvolvimento de um Cliente OPC DA

Como nota preliminar, deve referir-se que todas as aplicações que serão apresentadas de seguida, foram testadas em ambiente real, uma vez que foi utilizado um autómato programável igual e com as mesmas cartas que o utilizado na máquina de comprovar portas.

De seguida, serão descritas as várias fases do desenvolvimento do cliente OPC até se chegar à versão final pretendida, justificando-se as várias opções que foram sendo efectuadas.

Mais à frente serão descritas as alterações que é necessário introduzir no programa do autómato que controla a máquina, assim como o programa utilizado no autómato de testes para simular o normal funcionamento da máquina.

6.1.1 Configurações – Registo da Biblioteca do OPC no Windows

A primeira tarefa a realizar para começar a desenvolver a aplicação é preparar o sistema operativo para operar com a tecnologia OPC, é necessário proceder ao registo da biblioteca (DLL) do OPC, recorrendo para isso ao *Registry* do *Windows*.

O registo é executado da seguinte maneira:
1º Copia-se o ficheiro **OPCDAAUTO.DLL** para a directoria
C:\WINNT\System32\opddaauto
2º Registar o ficheiro como
C:\>regsvr32 C:\WINNT\System32\opddaauto

O passo a seguir teve por objectivo garantir a comunicação entre o autómato (CJ1) e o PC via *ethernet* tendo sido utilizado o software *Cx.-Programmer*.

De seguida, foi instalado e configurado o servidor OPC DA, tendo sido utilizado um software fornecido pelo próprio fabricante do autómato programável (aplicação *CX-Server OPC* da OMRON). Seguidamente escolhemos quais as posições de memória utilizadas no autómato para o controlo do processo e definimo-las como Itens no servidor. A partir deste momento todas as variáveis relevantes para conhecer o desempenho da máquina de comprovar portas passam a estar mapeadas no servidor OPC, e portanto disponíveis para serem recolhidas e tratadas pela aplicação cliente OPC que será desenvolvida.

Ainda nesta fase foi também realizado um teste para comprovar que o servidor estava a funcionar, que consistia na leitura de zonas de memória.

6.1.2 Desenvolvimento de um Cliente OPC para Trabalhar com um Servidor OPC DA

Neste momento, estamos em condições para iniciar o desenvolvimento de um cliente OPC que interactue com qualquer servidor OPC DA, desenvolvido utilizando o interface *OPC Automation* e utilizando como linguagem de programação o *Visual Basic 6*.

A aplicação desenvolvida apresenta um interface básico, simples e oferece as seguintes funcionalidades:

- Procurar os servidores OPC que estejam registados na máquina onde corre a aplicação.
- Seleccionar entre os servidores instalados.
- Criar um grupo no servidor e configurar o seu *Update Rate*.
- Realizar a conexão em função das configurações pretendidas.
- Activar e desactivar o grupo.
- Indicar o caminho para aceder aos itens o "*Fully qualified item ID*" (**Item ID**).

- Efectuar leituras de 5 itens por subscrição (**Auto Leitura ON**).
- Efectuar leituras e escritas síncronas de itens.

Figura 5.11 - Interface do cliente OPC desenvolvido para teste

Descrição do funcionamento

A aplicação ao ser iniciada vai automaticamente procurar os servidores OPC activos, e permitir que o operador seleccione um servidor, assim com adicione o nome e o *Update Rate* para o grupo de itens que serão consultados.

Para iniciar a recolha de dados o operador têm que conhecer os "***Fully qualified item ID***" de cada variável e digita-los nas caixas de texto "**Item ID**".

Depois de escolhidas estas configurações selecciona-se o botão "**Conectar**" (quando esta conectado apresenta a palavra "**Desconectar**"), neste momento a aplicação está pronta para começar a recolha de dados. A aplicação emite uma mensagem de erro se não for possível a conexão ao servidor e coloca a palavra erro no campo valor se não for possível recolher os dados de um dos itens.

Temos à disposição três hipóteses para a troca de dados, se carregarmos no botão "**Auto Leitura ON**" dá-se inicio a leituras dos itens por subscrição, se carregarmos no botão "**Leitura Síncrona**" e devolvido o valor actual dos cincos Itens, e se for seleccionado "**Escrita Síncrona**" e alterado o valor dos cinco itens (exp. cinco variáveis no autómato) com os valores que foram colocados previamente nas caixas de texto "**Valor**".

Relativamente a funções relevantes na programação no que diz respeito ao OPC temos:

- Procura de servidores OPC registados.
- Realiza a conexão e desconexão ao servidor OPC escolhido.
- Cria um ficheiro que guarda as configurações que a aplicação têm ao sair desta.
- Carrega o ficheiro de configurações ao iniciar a aplicação.
- Cria grupos de itens(*tags*) e adiciona itens(*tags*) a esses grupos.
- Utiliza a função *DataChange* que permite a leitura por subscrição.
- Permite activar e desactivar a leitura por subscrição.
- Efectua a leitura síncrona de cada item independente.
- Efectua a escrita síncrona de cada item independente.
- Detecta a falta ou falha do servidor OPC.
- Efectua o tratamento de erros ao nível da conexão e inclusão de itens e carregamento do ficheiro de configurações.

Esta aplicação foi testada em 4 servidores OPC (*Omron /Keepware/Matrikon/Schneider*), tendo atingido por completo os objectivos propostos.

Os itens utilizados nos testes apresentavam o conteúdo de zonas de memória do autómato CJ1 CPU12-ETH que eram incrementadas com frequência de 20ms, 100ms, 200ms,1000ms, em Decimal e BCD.

6.1.3 Registo dos Dados em Base de Bados

Partindo da aplicação de teste, foi incluído código que permite registar numa base de dados os valores provenientes do servidor OPC.

A aplicação foi implementada na linguagem *Visual Basic 6* e recorre a objectos ADO para aceder e manipular os dados numa base de dado *Access*. O **ADO** (*Activex Data Object*) é um interface de alto nível para acesso a bases de dados que pode ser utilizado em aplicações convencionais como em aplicações *web* e que permite interagir com vários tipos de fontes de dados como bases de dados, folhas de cálculo e ficheiros de texto. A possibilidade de interagir com vários tipos de fontes de dados permite, numa fase de desenvolvimento trabalhar com *Access* e, posteriormente, na fase de produção passar, por exemplo, para o gestor de base de dados *SQL Server*. Para isso, apenas é necessário alterar o *Provider* (OLEDB para ODBC) para o *Provider* (OLEDB para SQL Server).O **DSN** (*Data Source Name*) permite configurar o nome da base de dados, o endereço de servidor e o *driver* ODBC. Na aplicação desenvolvida, a base de dados é identificada através do DSN. Logo, qualquer alteração na localização da base de dados passa a ser reflectida apenas no DSN e não no código da aplicação.

A figura seguinte mostra a interface com o utilizador da aplicação desenvolvida nesta fase.

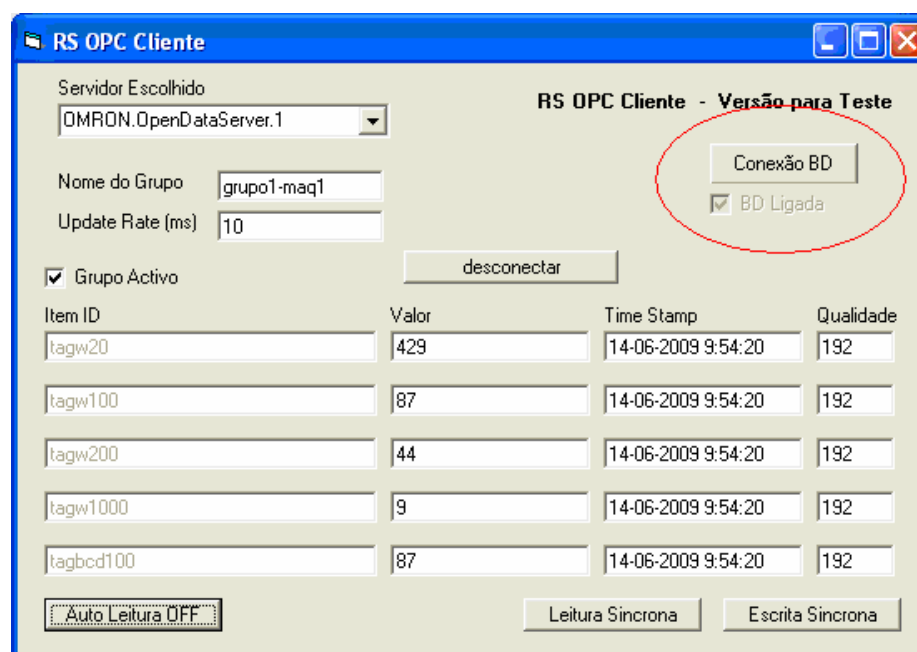
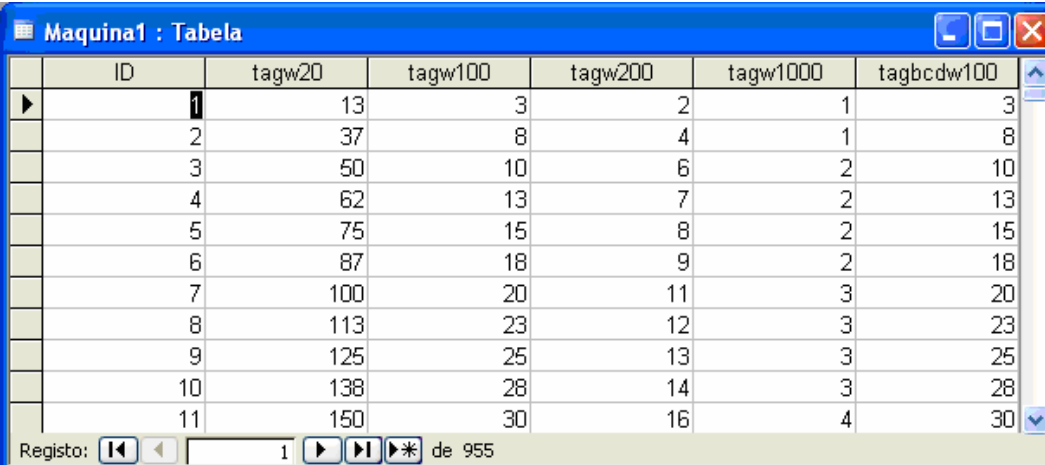



Figura 5.12 - Interface do cliente OPC com registo de dados na BD

Apresenta as seguintes funcionalidades:

- Procurar os servidores OPC que estejam registados na máquina onde corre a aplicação.
- Permitir a selecção entre os servidores instalados.
- Permitir criar um grupo no servidor e configurar o seu *Update Rate*.
- Realizar a conexão em função das configurações pretendidas.
- Activar e desactivar o grupo.
- Indicar o caminho para aceder aos itens o “*Fully qualified item ID*” (Item ID).
- Efectuar leituras de 5 itens por subscrição (Auto Leitura ON).
- Efectuar leituras e escritas síncronas de 5 itens.
- Activar ou desactivar o registo na base de dados.
- Informar se está activo o registo de dados.

Descrição do funcionamento

A aplicação funciona em todo igual a primeira versão, à excepção que ao ser iniciada activa o registo de dados na base de dados, posteriormente se necessário o operador pode desactivar esta função. Existe uma *CheckBox* que indica quando o registo de dados está activo ou não.

Relativamente a funções relevantes na programação temos:

- Procura de servidores OPC registados.
- Realiza a conexão e desconexão ao servidor OPC escolhido.
- Cria um ficheiro que guarda as configurações que a aplicação têm ao sair desta.
- Carrega o ficheiro de configurações ao iniciar a aplicação.
- Cria grupos de itens(*tags*) e adiciona itens(*tags*) a esses grupos.
- Utiliza a função *DataChange* que permite a leitura por subscrição.
- Permite activar e desactivar a leitura por subscrição.
- Efectua a leitura síncrona de cada item independente.
- Efectua a escrita síncrona de cada item independente.
- Detecta a falta ou falha do servidor OPC.
- Efectua o tratamento de erros ao nível da conexão e inclusão de itens e carregamento do ficheiro de configurações.
- Estabelece a conexão à base de dados *Access*.
- Grava os valores dos itens na base de dados.
- Itens com nível de qualidade fraca regista “Erro”.
- Efectua o tratamento de erros na conexão á base de dados.

A aplicação foi testada em vários servidores OPC (*Omron /Keepware/Matrikon*), em todos atingiu por completo os objectivos. Tal como no caso anterior, os itens utilizados nos testes apresentavam o conteúdo de zonas de memória do autómato CJ1 CPU12-ETH que eram incrementadas com frequência de 20ms, 100ms, 200ms,1000ms, em Decimal e BCD.

Seguidamente apresenta-se uma Figura onde se pode verificar o registo de valores na tabela Maquina 1 por parte da aplicação:

Maquina1 : Tabela

ID	tagw20	tagw100	tagw200	tagw1000	tagbcdw100
1	13	3	2	1	3
2	37	8	4	1	8
3	50	10	6	2	10
4	62	13	7	2	13
5	75	15	8	2	15
6	87	18	9	2	18
7	100	20	11	3	20
8	113	23	12	3	23
9	125	25	13	3	25
10	138	28	14	3	28
11	150	30	16	4	30

Registo: 1 de 955

Figura 5.13 - Imagem da Tabela1 criada pelo cliente OPC

6.1.4 Resultados e Conclusões

No fim deste sub-capítulo passamos a:

- Saber configurar um sistema para operar com a tecnologia OPC.
- Saber desenvolver um cliente OPC capaz de operar em qualquer servidor OPC e registar as suas transacções em bases de dados.

A principal conclusão deste tópico, foi o desenvolvimento um cliente DA que funciona de forma estável em vários servidores, e que permite o registo de dados.

6.2 Teste para Determinação do Mínimo Update Rate

Os testes apresentados de seguida permitem avaliar a performance da tecnologia OPC e sustentar opções de projecto com garantias de exequibilidade para atingir os objectivos pretendidos para o sistema de informação.

O primeiro teste realizado teve por objectivo determinar o mínimo *Update Rate*, ou seja determinar qual seria o período mínimo de variação de uma variável que um cliente era capaz de detectar e registar, usando o servidor OPC da *Omron* e o servidor da *Kepware*.

O fluxo de dados implementado no teste está representado na figura seguinte:

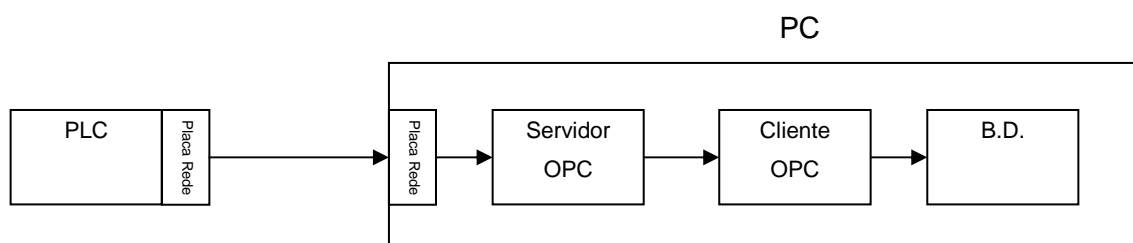



Figura 6.1 – Cadeia de dados para o teste de desempenho

Este fluxo de dados é idêntico ao que será implementado para a recolha de dados nas máquinas seleccionadas para o projecto.

6.2.1 Procedimento de Teste

Os passos envolvidos n a realização do teste foram os seguintes:

1. Configurar o PLC para ter disponível dois tipos de comunicação, via série e via *ethernet*.

A possibilidade de ter disponível dois tipos de comunicação vai permitir ter um canal totalmente disponível para transferência de dados e outro para o controlo e visualização do estado do PLC. Não existe a possibilidade de o controlo interferir na troca de informação e portanto diminuir as performances.

2. Configurar no PLC o endereçamento, criar a tabela de I/0 e seleccionar os *Dip Switchs*

3. Desenvolver e carregar o programa para o PLC:

O programa consiste em ter 5 variáveis que são incrementadas durante 60 segundos, com um pulso de relógio de 20ms, 100ms, 200ms e 1000ms. Todas as variáveis no início do teste estão nulas e ao fim de 60 segundos apresentam os seguintes conteúdos 3000, 600, 300 e 10 respectivamente. Se o cliente OPC conseguir acompanhar esta cadência. Teremos 3000, 600, 300 e 60 registos respectivamente ao fim de um minuto.

São utilizados incrementadores em binário e BCD que registam constantemente o seu conteúdo em células de memória do PLC.

É utilizado um *timer* de alta precisão para garantir com segurança a mesma contagem de tempo entre testes. Garante um erro máximo de 10 ms entre execuções

4. Configurar o servidor OPC da *Omron*.

5. Configurar o servidor OPC da *Kepware*.

6. Configurar o cliente OPC de modo a operar de quatro maneiras diferentes:

6.2.2 Testes Efectuados

Consoante o decorrer do teste vamos ter as seguintes configurações principais no cliente:

- Teste 0: *Update rate* de 10ms (Tempo entre leituras do servidor ao dispositivo) / Sem conexão à base de dados
- Teste 1: *Update rate* de 10ms / Cria uma nova conexão à base de dados sempre que se quer registar um valor.
- Teste 2: *Update rate* de 10ms / Cria-se a conexão à base de dados no início da execução do programa, mantendo-se activa à espera de instruções para o registo de dados.
- Teste 3: *Update rate* de 10ms / Igual ao teste anterior mas é adicionado ao servidor OPC quatro vezes mais carga.

Os testes, tiveram por objectivo geral avaliar o desempenho do OPC nas variadas situações, em relação à taxa de variação das variáveis (mínimo *update rate*).

Os objectivos específicos de cada um dos testes são detalhados de seguida:

Objectivo do teste 0: avaliar o mínimo *update rate* quando não realizamos o acesso à base de dados.
Consistiu numa observação cuidada da sequência do valor da variável. O valor que a variável vai tomando é observando no campo "valor" no cliente OPC. Foi-se gradualmente aumentando a taxa de variação até que a visualização da sequência do valor da variável fosse contínua.

Objectivo do teste 1, permite avaliar o mínimo *update rate* quando criamos uma conexão à BD cada vez que queremos registar um dado.
Consistiu em registar na base de dados, todos os valores das variáveis devolvidos pelo servidor ao cliente, em que para cada valor era criado uma conexão a base de dados e posteriormente realizado o registo.

Objectivo do teste 2, permite avaliar a utilização de dois canais de comunicação, a influência da capacidade de processamento, a variação do mínimo *update Rate* quando é implementado um controlo mais eficiente na base de dados.
Consistiu em registar na base de dados, todos os valores das variáveis devolvidos pelo servidor ao cliente, em que era criado uma conexão a base de dados que permanecia activa durante a execução do programa e que ia registando os dados em função da solicitação.
Foi executada uma variante onde aplicamos um canal totalmente disponível para transferência de dados (Comunicação via *ethernet*) e outro para o controlo e visualização do estado do PLC (Comunicação via Porta Série). Outra situação testada foi o aumento da capacidade de Processamento, passou-se de um processador *Pentium 4* (3,2 GHz / Cache de 2Mb / 1GB de Ram) para *Corel2 Duo T7200* (2 GHz / Cache de 4Mb / 2GB de Ram). Foi também comparado o desempenho entre servidores diferentes.

Objectivo do teste 3, Permitir avaliar a influência no registo dos dados perante o aumento de grupos e itens no servidor OPC.
Este teste desenvolveu-se dentro dos moldes do teste anterior, apenas aumentando a carga no servidor quatro vezes.

Foram realizadas oito séries (duas para o teste1(**T1**), 4 para o teste2 (**T2**) e 2 para o teste3 (**T3**)), cada uma com configurações distintas tal como já foi já referido. Cada série foi executada seis vezes, e posteriormente calculado a média de cada série.

Uma série tinha uma duração de 60 segundos, durante o qual são registados na base de dados os valores das variáveis, no final da série acede-se a base de dados para contabilizar o número de registos obtidos

6.2.3 Resultados

Na figura seguinte são apresentados os resultados das 8 séries realizadas. Para garantirmos uma base de trabalho igual, foi utilizado sempre o computador com o processador *Pentium 4* (**P4**) á excepção da série em que foi utilizado o computador com o processador *Corel2 Duo*(**P7200**).

Para cada teste(**T1,T2,T3**) tivemos duas variantes, uma com a comunicação via **Ethernet**, outra por comunicação via **Série**.

No Teste2 (**T2**), apareceram mais duas variantes, uma para testar a influência de o mesmo canal de comunicação servir para comando e transferência de dados(**T2-Sér_Ether-P4**) e o segundo para testar a influencia da capacidade de processamento (**T2-Ethernet-P7200**)

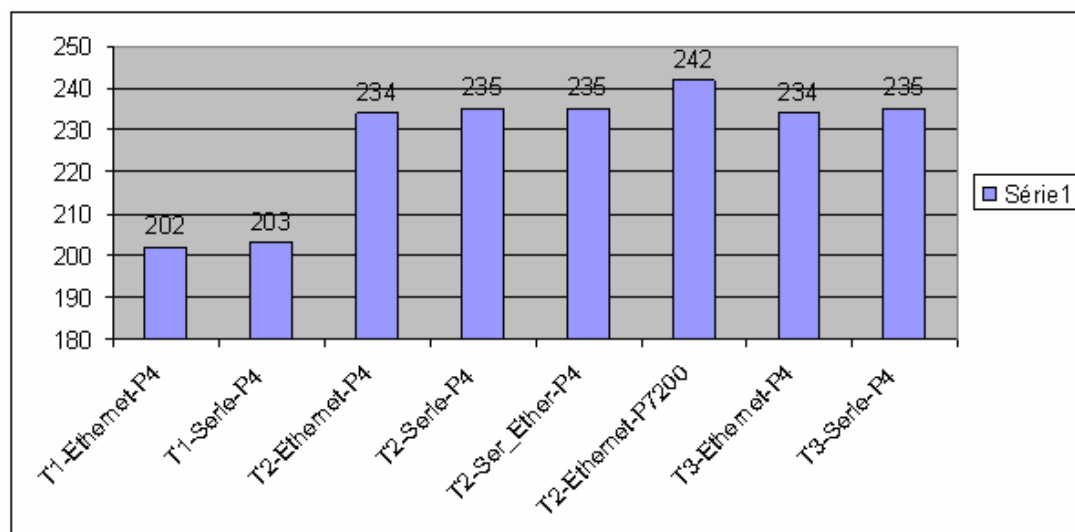



Figura 6.2 - Resultados do teste(nº de leituras por item em 60 segundos)

Foram também realizadas mais duas séries nas quais apenas foi utilizada uma variável com um pulso de relógio de 20ms. A diferença entre as séries residia nos servidores OPC utilizados.

T2-Ser_Ether-P4 – Utiliza o servidor da *Kepware*

T2- Ether_Ser -P4 – Utiliza o servidor da *Omron*

A comparação entre estas duas séries apresenta a diferença de desempenho dos servidores. Os resultados obtidos foram os seguintes:

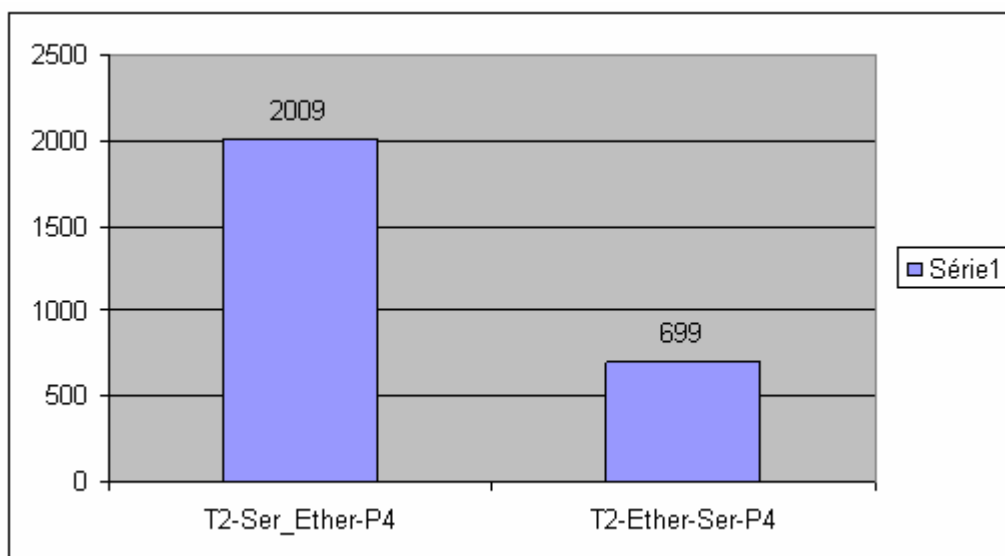



Figura 6.3 – Comparação entre os servidores da Kepware e Omron.

Para o Teste0, o resultado aproximado ronda os 250 ms, ou seja conseguimos visualizar uma variável que tenha uma taxa de variação de 250 ms. Para chegarmos a este valor o método consistiu em incrementar um valor a cada 200 ms, isso permitiu que na visualização da sequência em cada 10 registos, 2 eram perdidos.

6.2.4 Conclusões

Os testes efectuados permitiram retirar um conjunto de conclusões sobre OPC e o seu desempenho, as quais são expostas de seguida. Junto a cada conclusão, sempre que se achar pertinente, será apresentada uma justificação ou observação que a complemente.

1ª Conclusão: O facto da comunicação série apresentar, em todos os testes, mais um registo que na comunicação por *ethernet*, permite afirmar que não existe praticamente diferença no número de registos no mesmo espaço de tempo, entre a comunicação *Ethernet* e Série.

Este resultado justifica-se porque o acesso ao meio físico e o cumprimento de todos os procedimentos inerentes ao protocolo *ethernet* consomem tempo. No entanto, uma vez estabelecida a conexão, a comunicação via *ethernet* apresenta uma largura de banda muito superior à verificada na comunicação série. Ou seja para comunicações com grande volume de dados (mais itens e grupos) e portanto mais demoradas a *ethernet* passa a ter um desempenho superior.

Esta característica permite afirmar que, para sistemas que tenham poucas variáveis e um volume de dados pequeno a monitorizar, a transferência por série é uma escolha perfeitamente viável porque têm um desempenho muito bom e é mais económica que recorrer a comunicação *ethernet*, pois evita a necessidade de adicionar a carta de rede ao autómato.

2ª Conclusão: A tecnologia OPC apresentou nos testes realizados um grande estabilidade.

Esta conclusão é retirada ao analisarmos a variância dos resultados, que apresentaram em todas as séries dos testes (para uma mesma configuração) um máximo de 4 leituras de diferença entre o máximo e o mínimo

3ª Conclusão: A utilização de dois canais de comunicação, um para a transferência de dados e outro para o controlo, não vai variar o número de registos ao fim do teste.

4ª Conclusão: O aumento da capacidade de processamento não é proporcional aos resultados obtidos.

O PC com o processador *Core 2 duo* tem uma capacidade de processamento superior à diferença de 3,5% encontrada, quando comparamos os resultados do teste entre os dois computadores. Para se obterem dados mais rigorosos deveríamos ter utilizado um processador da mesma geração.

5ª Conclusão: Um algoritmo mais expedito e um controlo mais eficaz da base de dados permitiram aumentar em 17% a performance do sistema.

Os resultados obtidos estão justificadas com as alterações no algoritmo, relativamente a melhoramentos ao nível do OPC e a criar uma conexão que se mantêm activa durante a execução da aplicação em vez de ser criada uma nova cada vez que há um novo evento a registar.

6ª Conclusão: A interferência do acesso a base de dados representava aproximadamente 2%.
Este resultado foi obtido comparando os resultados do teste0 (sem acesso à base de dados) com os resultados do teste2 (com acesso à base de dados), e permite afirmar que o desempenho do sistema de recolha de registo dos dados não é praticamente influenciado pelo registo dos dados na base de dados.

7ª Conclusão: O servidor da *Kepware* é 2,87 vezes mais rápido que o da *Omron*.

Se houver uma grande necessidade de desempenho podemos sempre testar outros servidores, para ver qual o melhor que se adapta as necessidades. O servidor da *Kepware*

permite registar variáreis com um mínimo *update rate* de 30ms, que em comparação com os 250ms do servidor da Omron, é significativo.

A aplicação do servidor da *Kepware* é uma boa solução quando necessitamos de desempenho, mas menos robusta por que é susceptível de falhas, no decorrer dos testes várias vezes a base de dados apresentava erros nos registos e noutras situações no meio do teste parava de registar. Estas falhas nunca ocorreram com o servidor da *Omron*.

8ª Conclusão: Não foi possível retirar uma conclusão relativamente ao aumento de carga no servidor, uma vez que mesmo quadruplicando a carga os resultados mantinham-se iguais. Esta situação motivou novos esforços para responder a este pergunta.

6.3 Estudo de Performance Realizado pela OPC Foundation

O teste realizado anteriormente não permitiu tirar conclusões convincentes relativamente a aspectos como o aumento significativo grupos e itens no servidor.

Para complementar os resultados apresentados no ponto anterior e baseados nos nossos próprios testes, foram também analisados os resultados do estudo desenvolvido por Frank Iwanitz e Jürgen Lange sobe a alçada da *OPC Foundation* [Iwa06]. Trata-se de uma referência e está entre os mais completos estudos desenvolvidos para a especificação OPC DA OPC.

A análise cuidada dos resultados desse estudo, permitiram-nos retirar um conjunto de novas conclusões relevantes para a aplicação em estudo.

O *setup* para o estudo de performance consistiu em:

- Utilizar 2 *Pentium II* com uma frequência de 200 MHz, 1 *Pentium II* com uma frequência de 400 MHz e 1 com *Pentium II* com uma frequência de 550 MHz.
- Utilizar a plataforma *Windows NT*, e um cliente e servidor *Data Access*
- As medições foram executadas segundo as seguintes configurações: (A) - Cliente e servidor correm no mesmo PC (200MHz, 400MHz, 550MHz) e (B) – O cliente e servidor correm em maquinas separadas (dois PC de 200MHz) conectados por uma rede *ethernet* a 10 Mbit/s.

Cada série de medições foi executada da seguinte forma:

- Os itens (Objectos *OPCItems*) são incrementados em lotes de 400 começando no 0 até aos 20000 itens.
- As variáveis (itens) são do tipo Booleano, e são alteradas por um simulador.
- Os valores das variáveis são colocados na memória do PC através do servidor DA
- Durante os testes, os periféricos dos PCs não estão acessíveis.
- O numero de grupos criados (Objectos *OPCGroup*) foi dividido em quatro categorias (1, 10, 50 e 100 Grupos) para analisar o peso do numero de grupos no ensaio.
- Foi utilizada a leitura por subscrição(só quando ocorrer uma mudança numa variável é que o dados são transferidos.
- A taxa de variação dos itens é simulada de forma a termos ao mesmo tempo 5%,10%, 50% ou 100% de variações simultâneas.

No início do teste no instante t=0 são lidos os valores de todos os itens, seguidamente o cliente cria os grupos e começa a aumentar gradualmente o número de itens, dividindo-os de forma igual pelos grupos. Continuamente são medidos e registados os tempos das transacções (tempo desde o pedido até que toda a operação esteja concluída), calculado posteriormente a média de *Update Rate* para cada grupo, em função das mudanças simultâneas (da carga).

Cada série de teste teve duas variantes, o servidor e cliente na mesma máquina e em máquinas separadas.

Os resultados do estudo estão apresentados no Anexo C.

6.3.1 Conclusões

As conclusões que se seguem foram retiradas com base em conclusões e na análise cuidada dos resultados do estudo, recorrendo para isso a interpolações.

1 - Quando temos 20000 itens, e uma troca simultânea de 5% (1000 itens), o servidor e cliente na mesma máquina:

A) A influência do número de grupos é visível mas não é significativa.

Tabela 6.1 – Conclusão dos resultados da série 1

Configuração	Nº de grupos	Carga	Tempo (ms)
Mesma Máquina-COM	100	5%(1000 itens)	621
Mesma Máquina-COM	1	5%(1000 itens)	524

B)A diferença entre 1 grupo e 10 grupos é praticamente nula.

2 - Quando temos 20000 itens, um troca simultânea de 5% (1000 itens), o cliente e servidor em PCs separados:

A)A influência do número de grupos é visível mas não é significativa

Tabela 6.2 – Conclusão dos resultados da série 2

Configuração	Nº de grupos	Carga	Tempo (ms)
Máquinas Separadas-Dcom	100	5%(1000 itens)	719
Máquinas Separadas-Dcom	1	5%(1000 itens)	584

B)A diferença entre 1 grupo e 10 grupos é praticamente nula.

3 – Se compararmos as séries anteriores vemos que as diferenças não são significativas, e como é natural são uma consequência do uso da rede, se considerarmos 50 grupos temos uma diferença de 114ms.

Tabela 6.3 – Conclusão dos resultados da série 3

Configuração	Nº de grupos	Carga	Tempo (ms)
Máquinas Separadas-Dcom	50	5%(1000 itens)	638
Máquinas Separadas-Dcom	1	5%(1000 itens)	584
Mesma Máquina-COM	50	5%(1000 itens)	566
Mesma Máquina-COM	1	5%(1000 itens)	524

4 – Quando temos 20000 itens, 10 grupos e uma carga variável e o servidor e cliente na mesma máquina:

A) A influência da carga é significativa, porque o mínimo *Update Rate* para uma carga de 100% é sensivelmente 2 vezes o mínimo *Update Rate* para uma carga de 0%.

Tabela 6.4 – Conclusão dos resultados da série 4

Configuração	Nº de grupos	Carga	Tempo (ms)
Mesma Máquina-COM	10	100%	1008
Mesma Máquina-COM	10	50%	770
Mesma Máquina-COM	10	10%	560
Mesma Máquina-COM	10	5%	545
Mesma Máquina-COM	10	0%	498

B) Até 10% da carga não temos uma variação significativa do mínimo *Update Rate*.

C)Temos uma velocidade de transferência de dados de 3,8Mbits/s, quando temos a carga de 100%.

D)Mesmo quando não temos variação de itens (carga 0%) o sistema contínua em constante operação (o valor não é recebido mas tem que ser verificado), sendo esta a razão porque temos neste caso 498ms de *Update Rate*.

5 – Quando temos 20000 itens, uma carga variável e o cliente e servidor em PCs separados:

A) A influência da carga é bastante significativa, porque o mínimo *Update Rate* para uma carga de 100% é sensivelmente 4,5 vezes o mínimo *Update Rate* para uma carga de 0%.

Tabela 6.5 – Conclusão dos resultados da série 5

Configuração	Nº de grupos	Carga	Tempo (ms)
Máquinas Separadas-Dcom	10	100%	2315
Máquinas Separadas-Dcom	10	50%	1409
Máquinas Separadas-Dcom	10	10%	675
Máquinas Separadas-Dcom	10	5%	590
Máquinas Separadas-Dcom	10	0%	498

B) Temos uma velocidade de transferência de dados de 1,67Mbits/s, quando temos a carga de 100%. A velocidade standard utilizada em redes nos dias que correm situa-se nos 100Mbits/s, perante a velocidade de transferência de dados já determinada (1,67Mbits/s), verifica-se que sobra muita largura de banda na rede para outras aplicações/funções.

6- Se compararmos as duas séries anteriores vemos que as diferenças de acentuaram significativamente por duas razões, o aumento da carga que agora se situava nos 100% e a consequência do uso da rede.

Tabela 6.6 – Conclusão dos resultados da série 6

Configuração	Nº de grupos	Carga	Tempo (ms)
Máquinas Separadas-Dcom	10	100%	2315
Mesma Máquina-COM	10	100%	1008
Máquinas Separadas-Dcom	10	10%	675
Mesma Máquina-COM	10	10%	560

Com uma carga de 10 % (2000 itens) não é significativa a diferença, mas para uma carga de 100% já é superior ao dobro e tende a aumentar rapidamente em função do volume de dados.

7- Quando avaliamos a influencia da velocidade do processador no mínimo Update Rate verificamos que:

A) A duplicação da frequência de relógio (200MHz para 400MHz) quase reduz para metade o mínimo *Update Rate*.

B) Utilizando o PC a 550MHz obtivemos uma redução de 25% no tempo face ao processadora 400MHz.

Tabela 6.7 – Conclusão dos resultados da série 7

Configuração	Nº de grupos	Carga	Tempo (ms)
PC 550MHz	50	100%	227
PC 400MHz	50	100%	299
PC 200MHz	50	100%	580

C) O aumento da capacidade de processamento pode ser uma solução para resolver problemas de desempenho.

É necessário ter em conta que para diferentes trocas de dados (existem três tipos), vamos ter resultados diferentes. Para este estudo cada vez que queremos ler os 20000 itens temos de trocar 480.2Kbytes de informação. A figura seguinte apresenta a quantidade de dados de uma transmissão entre o servidor e o cliente.

TransactionIdentifier	4 byte
GroupHandle	4 byte
MasterQuality	4 byte
MasterError	4 byte
Number of OPCItem objects in the call per OPCItem object	4 byte
ClientHandle	4 byte
Value	6 byte (4 byte value + 2 byte type information)
Quality	2 byte
TimeStamp	8 byte
Error	4 byte
All together	480,200 byte for 20,000 OPCItem objects

Figura 6.4 - Quantidade de dados de uma transmissão entre o servidor e o cliente

Outros estudos realizados comparando DCOM com XML demonstraram que:

A utilização de servidores OPC XML DA (utilizam XML para realizarem a troca de informação em vez de DCOM, pode ser utilizado em diferentes plataformas) são bastante mais lentos que os servidores OPC DCOM DA, logo não devem ser utilizados em aplicações que necessitem de um elevado desempenho. Em termos de comparação num teste onde eram lidos 1000 itens, o mínimo *update rate* do servidor OPC DCOM DA e aproximadamente 6,5 vezes mais rápido que o servidor OPC XML DA.

A utilização das platafomas *Linux* ou *Windows* não apresenta vantagens em termos de desempenho, A plataforma *Linux* é mais rápida 3,5%.

As diferenças entre utilizarmos um acesso via *Internet* ou *Intranet*, situa-se nos 30% a pender naturalmente para o lado da *Intranet*. Estes dados demonstram que é também uma solução o desenvolvimento de aplicações que troquem dados via *Internet*.

6.4 Aplicabilidade da Tecnologia OPC

Na avaliação da viabilidade da utilização da tecnologia OPC no sistema em estudo, deve ter-se em conta, além dos resultados dos estudos anteriores, que estamos perante uma linha de montagem contendo cerca de 30 máquinas, com ciclos de funcionamento de 8 a 10 segundos e que registam, em média e por ciclo de funcionamento, 3 registos, correspondentes aos eventos associados à transições no diagrama de estados das máquinas, conforme discutido no capítulo 2.

Os cálculos permitem concluir que é possível ter aproximadamente 675 registos por minuto (7,5 ciclos por minuto*3 eventos por ciclo*30máquinas) na linha, consoante o tipo de leitura a utilizar podemos ter 30 ou 60 grupos (se tiver variáveis de controlo) e teremos 240 eventos (8estados*30máquinas) diferentes. Se pensarmos que termos de usar 2 *Dwords* (será justificado mais a frente) para registar um evento, teremos então 480 Itens (1 item por *DWord*). Estamos perante valores estimados uma vez que não foram estudadas de forma minuciosa todas as máquinas da linha, mas admitiu-se terem um comportamento idêntico, situação que em larga maioria se verifica.

Comparando os resultados desta análise, verificamos que, ao serem cruzados com os estudos anteriores, conclui-se que a configuração do primeiro teste permitia 1175 (235*5) registos por minuto o que permite ter uma margem de segurança confortável se considerarem aproximadamente 700 registos possíveis da linha.

Considerando, agora, o estudo da *OPC Foundation*, e fazendo uma interpolação aos dados do gráfico da figura C5 na linha "DCOM – 10 Groups 10%", obtemos, que para a pior situação (cliente e servidor em máquinas separadas - situação que não se verifica neste projecto), é possível ter como mínimo *update rate* aproximadamente 38,46ms, isto quer dizer que em cada 38,46 ms podemos registar 48 itens (10% dos 480Itens).

Comparando este valor com os 675 registos por minuto (7,5 ciclos por minuto*3 eventos por ciclo*30máquinas) vezes 2 itens necessários para cada registo temos 1350 itens por minuto lidos. Eram necessários apenas 1082 ms para ler os possíveis 675 registos que a linha pode produzir no máximo num minuto.

As conclusões anteriores permitem afirmar com relativa segurança que uma solução baseada em OPC satisfaz os requisitos do sistema de informação em estudo.

Capítulo 7

Implementação do Sistema de Informação

Descrever o sistema implementado com o objectivo de satisfazer os requisitos identificados no capítulo 2 e recorrendo às tecnologias estudadas nos capítulos 4, 5 e 6.
O sistema desenvolvido está organizado nos seguintes módulos:

- Modulo de aquisição de dados - Aplicação que efectua a recolha e registo de dados e que se subdivide em duas partes. A primeira parte opera as máquinas controladas por PLCs, e é aplicada a tecnologia OPC, a segunda parte opera as máquinas controladas por IPCs e é aplicada a tecnologia DCOM.
- Módulo de tratamento de dados – Aplicação que efectua o tratamento dos dados e apresenta os indicadores para o sistema de informação. Recorre a base de dados criada pelo módulo de aquisição de dados para efectuar o tratamento dos dados e apresentar os indicadores.

A organização do capítulo é a seguinte:

- No ponto 7.1 são analisados aspectos para a implementação do sistema de informação.
- No ponto 7.2 é apresentada a arquitectura do sistema de informação e descrito como o fluxo de dados e a organização das camadas do modelo para o sistema são implementados, com especial relevância na recolha de dados via OPC e DCOM. No último sub ponto é apresentada a arquitectura das camadas para o módulo de aquisição de dados
- No ponto 7.3 é apresentada a aplicação que corre nos controladores locais de cada máquina (autómatos programáveis) e que é responsável pelo registo dos dados numa

fase primária no PLC, para que posteriormente o servidor OPC os recolha. É também definido a constituição de um registo para que a informação proveniente das máquinas seja estandardizada.

- No ponto 7.4 é apresentada a base de dados.
- No ponto 7.5 é apresentado o desenvolvimento do módulo de aquisição de dados. São desenvolvidos dois tipos de clientes (síncrono e assíncrono) e analisadas as suas vantagens e desvantagens em função do sistema a implementar, é então escolhido o cliente síncrono e apresentada a aplicação final testada na máquina.
- No ponto 7.6 é apresentada uma aplicação a pensar num teste para toda a linha.
- No ponto 7.7 são apresentadas as conclusões em função dos resultados obtidos

7.1 Análise/Aspectos para a Implementação do Sistema de Informação

Conforme foi já exposto em capítulos anteriores, optou-se pela especificação OPC DA para a transmissão de dados entre o servidor e o cliente OPC.

Esta especificação funciona com base no paradigma de memória partilhada. Como tal, a mudança de uma variável pode não ser registada se for escrito um novo valor antes do anterior ter sido lido e registado, o que pode ocorrer quando o período de amostragem é pequeno ou devido ao congestionamento da rede.

Mesmo quando há uma grande margem de segurança (“folga”) na transferência dos dados, podem acontecer situações de congestionamento da rede como por exemplo uma estação bloqueada ou uma estação com um funcionamento deficiente, em que o funcionamento sob o paradigma da memória partilhada provoca a perda de dados. Com a margem que temos, essa possibilidade era bastante remota e quase insignificante, mas deve ser considerada a possibilidade da solução ser aplicada em sistemas de maior dimensão, ou até mesmo em juntarmos as redes independentes para cada linha de montagem numa só (em estimativa 6 vezes maior).

No controlo de processos em tempo real, em que as acções futuras no processo são desencadeadas em função do estado actual das variáveis, não podemos recorrer a um leque de técnicas que nos permitem ultrapassar a condição da memória partilhada. No sistema que estamos a desenvolver, conhecer a evolução do processo do processo com 1 um minuto de atraso não é relevante, vão ser aplicadas técnicas (serão explicadas posteriormente em 7.3.1.) que permitam sobreporem-se ao paradigma da memória partilhada, garantindo assim que não aja perda dos dados.

7.2 Arquitectura do Sistema de Informação

Tomando em consideração a arquitectura geral definida na secção 3.1, que apresenta um computador por linha de produção, onde fica alojado a aplicação que controla o sistema de informação, e um computador central que recolhe a informação de cada linha.

Foi definida a arquitectura de controlo em termos específicos para o sistema de informação, que está apresentada na figura seguinte.

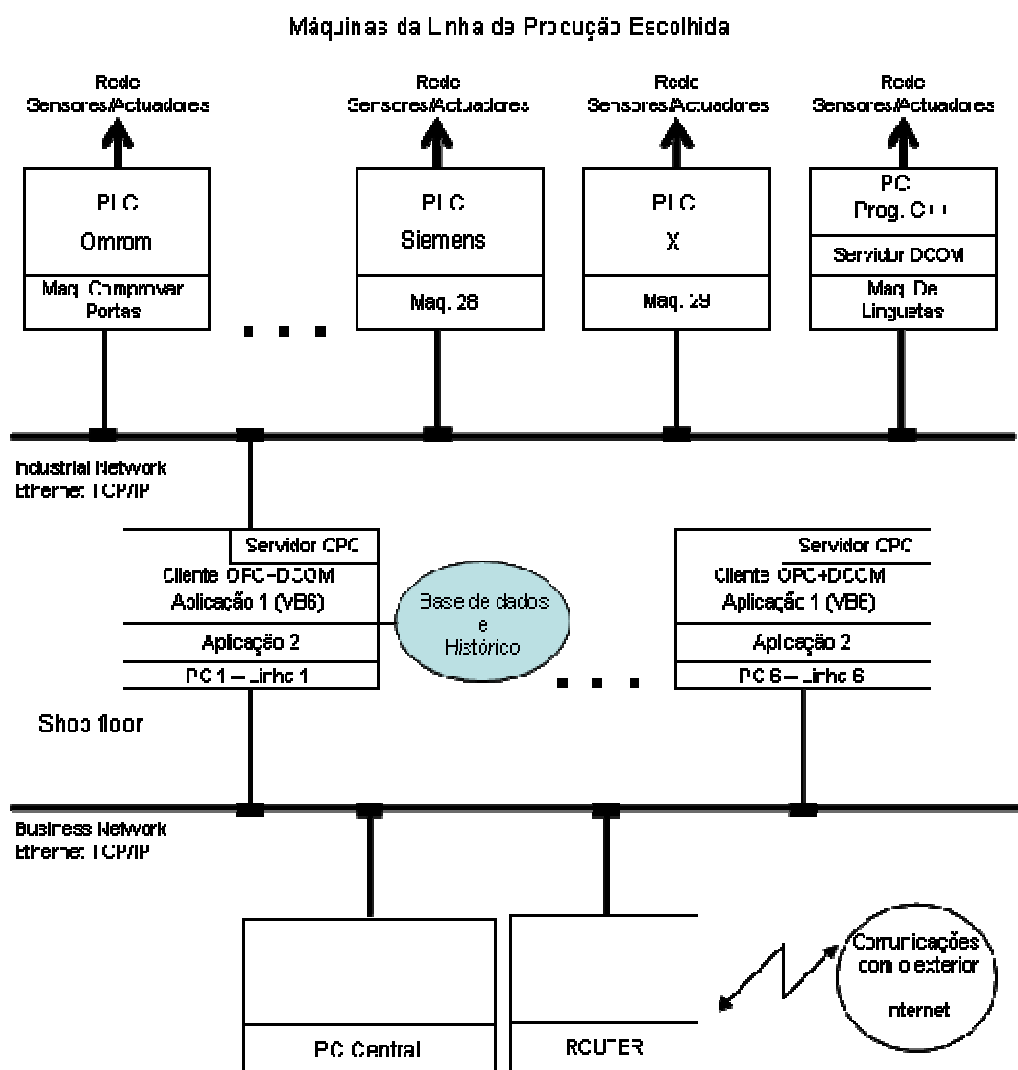



Figura 7.1 – Arquitectura de controlo do sistema

Analisando a figura vê-se, que o computador que controla o sistema de informação da linha 1 (PC1 – Linha1) recorre a dois módulos para cumprir os objectivos definidos para o sistema de informação e que estão apresentados no ponto 2.4, o módulo de aquisição de dados (aplicação 1) e módulo de tratamento de dados (aplicação 2).

O primeiro módulo é responsável pela recolha de dados provenientes das máquinas da linha e registo dos mesmos na base de dados. O segundo realiza o tratamento dos dados

recorrendo para isso ao acesso e preenchimento da base de dados e posterior apresentação de informação referente a linha como sejam os indicadores de desempenho. Também no computador da linha 1 serão instalados os servidores OPC que vão permitir a comunicação com os PLCs.

Na linha de produção onde se vai realizar o projecto, têm-se um especial relevo nas máquinas onde se vão realizar os testes (Máquina de comprovar portas e Máquina de linguetas), e onde se salienta o servidor DCOM (componente de software a adicionar à aplicação que controla a máquina) que fica situado no IPC da máquina de linguetas.

O que se verifica na linha 1 (a que foi escolhida para a realização do projecto), será repetido numa fase de implementação nas restantes 5.

Como já tínhamos definido e se pode verificar na figura anterior, a recolha de dados dos equipamentos pode-se fazer de duas maneiras diferentes, ou via OPC ou via DCOM. Serão explicadas posteriormente.

7.2.1 Fluxo de Dados e Organização das Camadas do Modelo para o Sistema de Informação

Na figura seguinte pode-se ver esquematicamente o fluxo de dados no sistema, permitindo assim ver como a informação circula no sistema de informação.

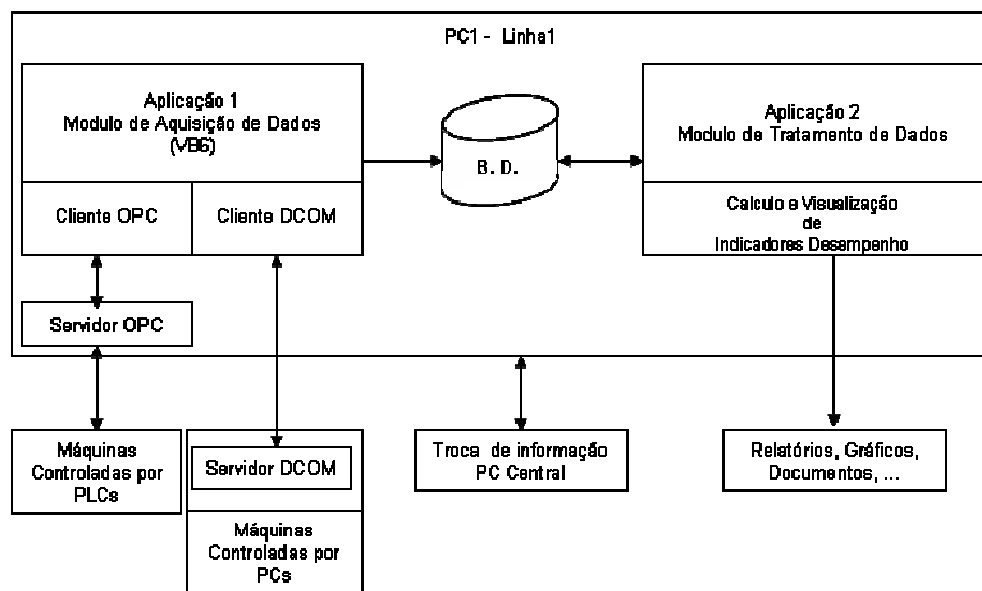



Figura 7.2 – Fluxo de dados do sistema

A organização das camadas que garante o controlo da arquitectura tem por base o seguinte modelo:

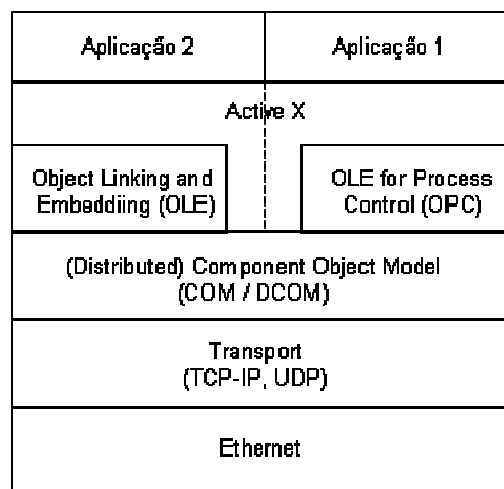



Figura 7.3 – Organização das camadas do modelo para o SI

7.2.2 Recolha via OPC

Como já foi justificado anteriormente no capitulo5, recorreu-se ao OPC para a recolha dos dados provenientes das máquinas controladas por PLCs, na figura seguinte podemos ver a configuração utilizada para recolher os dados da máquina de comprovar portas. Esta configuração será igual para as restantes máquinas controladas por PLCs.

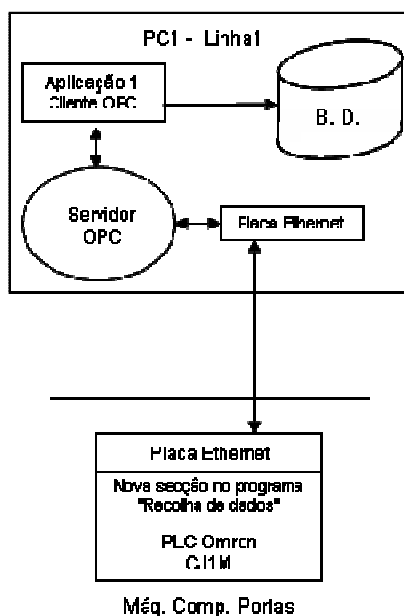



Figura 7.4 – Recolha via OPC

É importante referir que o programa do PLC vai necessitar de uma nova tarefa para recolher e disponibilizar os dados, e que o servidor OPC têm que ser configurado para ter acesso às variáveis do PLC pretendidas.

7.2.3 Recolha via DCOM

Relativamente a recolha por via DCOM, esta serve para recolher os dados de máquinas que utilizam um IPC e que têm uma aplicação dedicada que controla a máquina, esta abordagem esta justificada capitulo 4, na figura seguinte pode ver-se como será aplicada.

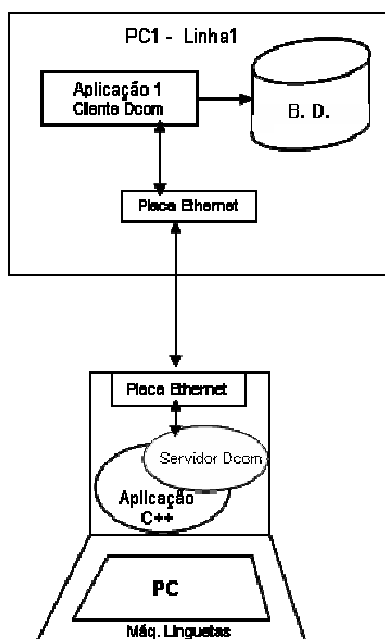



Figura 7.5 – Recolha via DCOM

Deve-se salientar que o Servidor DCOM é um componente DCOM que é adicionado a aplicação que controla a máquina e que recolhe e disponibiliza para o cliente DCOM os dados do ciclo de funcionamento da máquina.

7.2.4 Arquitectura das Camadas para o Modulo de Aquisição de Dados

Perante as duas possibilidades para a troca de informação, vamos ter a seguinte arquitectura das camadas para o módulo de aquisição de dados (aplicação1):

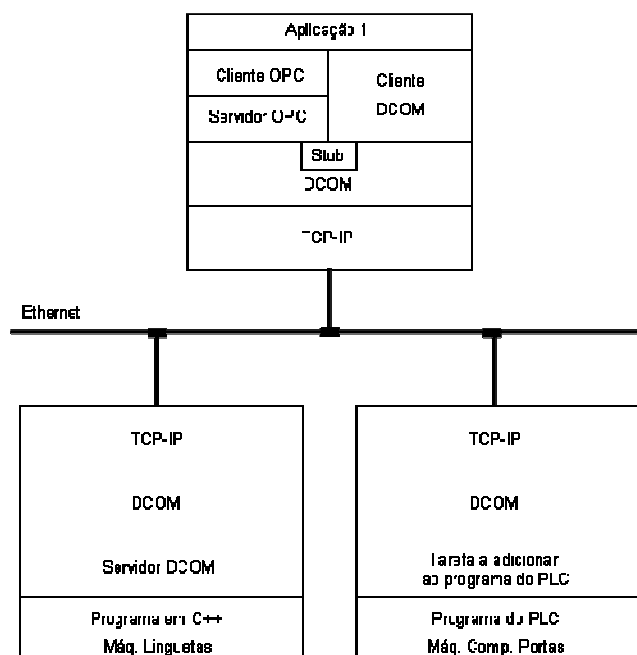



Figura 7.6 - Arquitectura das Camadas para o Modulo de Aquisição de Dados

7.3 Configurações na Máquina de Comprovar Portas (no PLC)

Desde que o PLC tenha memória disponível e não se apresente nos seus limites de processamento, é relativamente simples adicionar uma nova tarefa que seja executada em paralelo com as existentes, mesmo que o programa não seja aberto. Esta vantagem é muito interessante para o OPC uma vez que podemos recorrer a inclusão de tarefas que nos permitam aceder a informação que posteriormente será recolhida pelo Servidor OPC.

Como a *Huf* é quem projecta e acompanha o desenvolvimento das suas máquinas, mantêm uma relação com os integradores que lhes permite ter acesso aos programas que estão presentes nos autómatos. Não são assim de prever dificuldades na implementação desta solução mas, mesmo não tendo acesso ao programa do autómato, haveria duas soluções possíveis. A primeira seria identificar quais as zonas de memória que variavam com o início do estado e a segunda colocar uma carta e desenvolver um sistema paralelo capaz de recolher informação pretendida e disponibiliza-la ao servidor.

Uma outra possibilidade era pedir ao integrador a alteração do programa para passar a executar as funcionalidades secundárias.

7.3.1 Algoritmo da Tarefa de Registo Local dos Dados

Para responder ao paradigma da memória partilhada, foi necessário adicionar uma tarefa no autómato com o objectivo de registar os eventos correspondentes a informação dos instantes em

que temos as transições, primeiramente para um *buffer* e posteriormente em tempo oportuno para o cliente OPC.

A tarefa implementada no autómato reserva duas áreas distintas que funcionam como *buffers*, a primeira serve para registar os eventos a medida que ocorrem, a segundo serve como *buffer* de transferência.

A tarefa em termos específicos vai registar a ocorrência dos eventos no *buffer 1*, quando este se apresentar cheio é activa uma variável "activa" e transferido o conteúdo do *buffer* 1 para o *buffer* 2. O autómato vai continuar a registar os eventos naturalmente. Enquanto isto o servidor detecta por subscrição a mudança da variável e o cliente por consequência dá a ordem para se ler os itens correspondentes ao *buffer* 2, no final da operação volta a colocar a variável a zero.

A imagem seguinte demonstra de forma esquemática o algoritmo da tarefa a adicionar ao PLC:

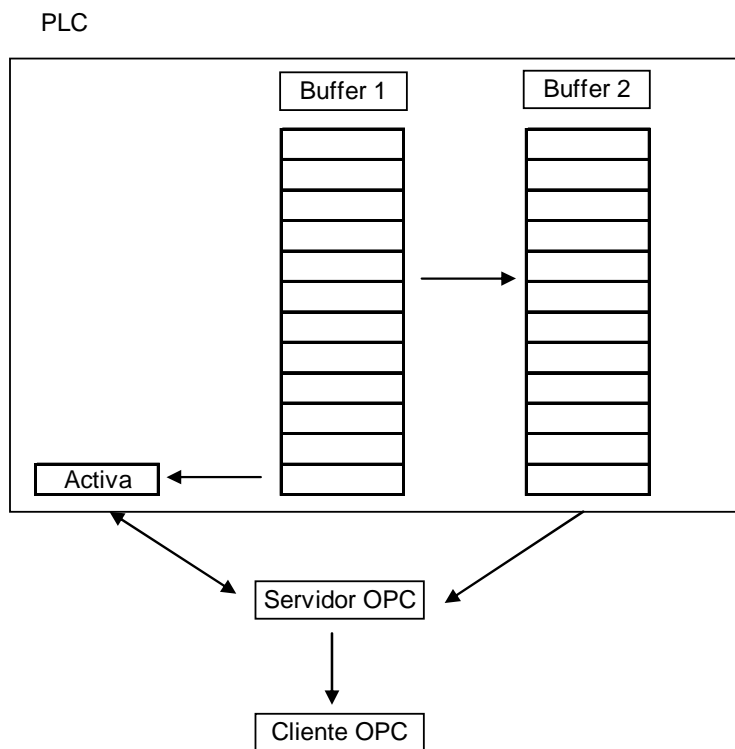



Figura 7.7 - Tarefa a adicionar ao PLC

No PLC foi implementado uma tarefa que criava uma tabela de registos e toda a programação capaz de responder de forma satisfatória aos objectivos pretendidos. É importante salientar que a tarefa estava prepara para que enquanto o *buffer* 2 não fosse registado, mesmo que o *buffer* 1 estivesse cheio não era apagado, ficava a partir deste momento a espera e o

último registo recebia informação a demonstrar que se estava a perder informação. Na figura seguinte apresenta-se o algoritmo da tarefa.

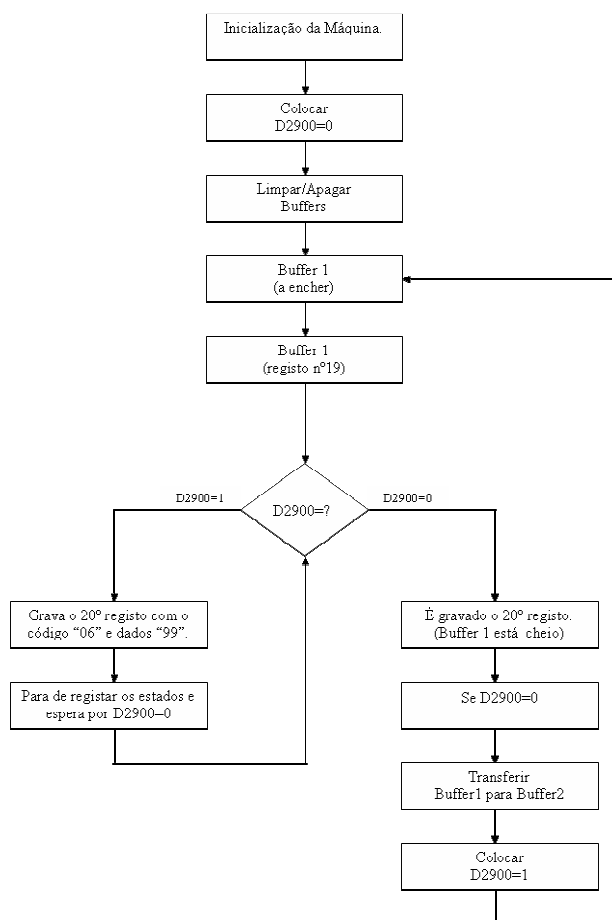



Figura 7.8 – Algoritmo da tarefa a adicionar ao PLC

A célula de memória D2900, assim como as células usadas nos *buffers* retêm informação mesmo com a falha de energia no PLC.

Deve-se ter em conta que a variável D2900 é colocada a zero pelo cliente OPC da aplicação 1 ou no arranque do PLC.

7.3.2 Qual o Relógio a Utilizar, para Adicionar o Instante Temporal aos Registos

Uma questão pertinente de se fazer nesta altura era, a utilização do relógio do autómato ou do relógio do sistema operativo do PC. A escolha recaiu no relógio do autómato, porque se vamos preencher o *buffer* o mais correcto é utilizar o relógio de tempo real que o autómato possui, no entanto se fizéssemos uma leitura directa da ocorrência dos eventos, temos que ter em conta que o instante temporal em que o corre o evento e outra e o instante temporal em que o servidor leu e registou o evento. Não será uma diferença muito grande mas existe, por esses motivo em casos deste tipo é preferível utilizar o relógio do dispositivo.

Para registarmos o tempo de forma a que possamos ter uma variável com o tempo standard é necessário recorrer a 2 *words*, como se verá na tabela 7.1.

7.3.3 Constituição de um Registo.

Cada registo possui o instante temporal em que à máquina entrou num dos seus estados, assim como informação sobre o respectivo estado ou possível erro que possa surgir.

O registo é constituído por 5 *words* estruturadas como na tabela seguinte, são registados em cada *Word* os campos seguintes:

Tabela 7.1 – Constituição de um Registo

Registo	Bits	
	15 8	7 0
1ª word	Hora (00 a 23)	Minuto (00 a 59)
2ª word	Segundo (00 a 59)	Décimo de segundo (0 a 9)
3ª word	Código do Estado	Dados (Ex. código do erro)
4ª word	-	-
5ª word	-	-

Como o relógio do PLC não disponibiliza décimos de segundo, foi necessário encontrar uma técnica que permitisse a obtenção deste campo. A solução consistiu em ter uma variável que era constantemente incrementada com uma frequência de 100ms entre de 0 a 9, voltando posteriormente ao início, e sempre que ocorre um evento também é registado também o valor desta variável.

O campo do código é utilizado para registar o número do estado em que a máquina entrou, servindo também para indicar a perda de registos quando os *buffers* estão cheios. O campo de dados serve para transportar informação relativa ao estado como seja a indicação do erro quando se tratar de um estado de erro.

A 4ª e 5ª *Word* foram deixadas livres prevendo futuras evoluções do sistema.

Os dados no registo são do tipo BCD, cada *buffer* tem uma capacidade de 100 *words*, o que perfaz 20 eventos. No caso da máquina em estudo um *buffer* é capaz de registar 7 ciclos do normal funcionamento da máquina o que significa uma autonomia de 70 segundos ou 140 segundos se considerarmos os dois *buffers*. O tamanho do *buffer* pode se necessário ser aumentado porque a programação esta feita a pensar nesta possibilidade, ao deixar variáveis estrategicamente colocadas que carregam por exemplo do tamanho da tabela de registos.

Os campos ano, mês, e dia são adicionados na aplicação cliente que controla o sistema de informação, recorrendo para isso à data do sistema operativo.

7.4 Base de Dados

A base de dados capaz de fornecer está informação terá uma tabela para todas as máquinas. Essa tabela terá os seguintes campos:

- ID - índice do registo.
- CodMaq – código da máquina onde ocorreu o evento
- CodActual - campo da base de dados onde é registado o código do estado actual da máquina.
- DadosErros – Campo onde é registado o tipo de erro que provocou a paragem da máquina. Este campo também tem a possibilidade de poder ser utilizado para registo de dados ou outra informação relevante para o processo, quando o estado actual não é de erro.
- TStamp – Campo onde é registado o instante temporal em que a máquina entrou no estado actual, é composto por horas, minutos e segundos.
- DSegundos – Campo utilizado para ser registado ao nível do décimo de segundo o instante temporal em que a máquina entrou no estado actual. Este campo justifica-se pelo simples facto de poderem suceder-se dois estado no mesmo segundo, nomeadamente quando se da inicio ao ciclo normal de funcionamento e a máquina já apresenta um erro.
- Dia – Campo onde é registado o dia da ocorrência do estado actual da máquina
- Mês – Campo onde é registado o mês da ocorrência do estado actual da máquina
- Ano - Campo onde é registado o ano da ocorrência do estado actual da máquina
- TempoCompleto - Campo onde é registado em formato regularizado o instante em que ocorreu o estado actual da máquina. Torna-se necessário este campo para que na fase de cálculo dos indicadores seja utilizado para cálculos temporais com recursos a funções predefinidas pela linguagem de programação. Este campo é composto pela data seguida da hora com a seguinte formatação “14-07-2008 16:09:23”
- CodAnterior – Campo que permite saber de forma directa, sem pesquisa o estado anterior da máquina. Torna-se importante este campo para evitar ocupação de recursos em futuras consultas a base de dados e também porque o ciclo das máquinas nem sempre é o mesmo.

Na escolha dos instantes e informação a adquirir/registar, foi tido em conta os dados recolhidos nas máquinas mais apropriadamente os diagramas de estados das mesmas. Ou seja com estes campos é possível registar todos os instantes temporais relevantes da máquina, conhecer o evento que os despoletou e, se necessário, adicionar informação a esse estado para, posteriormente, se proceder ao cálculo dos indicadores.

Posteriormente serão desenvolvidas tabelas indexadas a tabela das máquinas, tabelas essas que vão registar a informação referente a cálculos e indicadores obtidos, recorrendo para isso à consulta da tabela das máquinas.

O módulo de tratamento de dados fará a gestão da base de dados, o módulo de aquisição de dados regista apenas a informação referente a tabela primária das máquinas.

7.5 Módulo de Aquisição de Dados

Definidos e apresentados os aspectos de base para o sistema de informação, vai ser apresentado o módulo de aquisição de dados, que com já foi referido recolhe a informação proveniente das máquinas, e regista essa informação numa base de dados.

Inicialmente desenvolveu-se um cliente síncrono e um cliente assíncrono, muito próximos do módulo de aquisição final pretendido, depois de uma análise de ambos foi escolhido o cliente síncrono e realizadas as alterações finais e posteriormente executado o teste directamente na linha.

Deve-se salientar que todos os testes realizados ao software mesmo em fase de desenvolvimento eram realizados em ambiente real porque a cadeia de aquisição era igual a que seria encontrada na linha.

7.5.1 Desenvolvimento de Cliente OPC Síncrono

Vão ser adicionadas no cliente OPC DA desenvolvido no capítulo 6.1.3 mais funcionalidades, que vão ser executadas em paralelo com as já desenvolvidas. O objectivo fundamental desta aplicação é a recolha dos itens (registo dos eventos da máquina) recorrendo a uma leitura síncrona. Esta leitura é desencadeada quando a variável de controlo é activa (esta está localizada no autómato e significa *buffer* cheio).

A aplicação desenvolvida, apresenta o interface da figura seguinte, mantendo as capacidades da aplicação anterior e adicionando as seguintes:

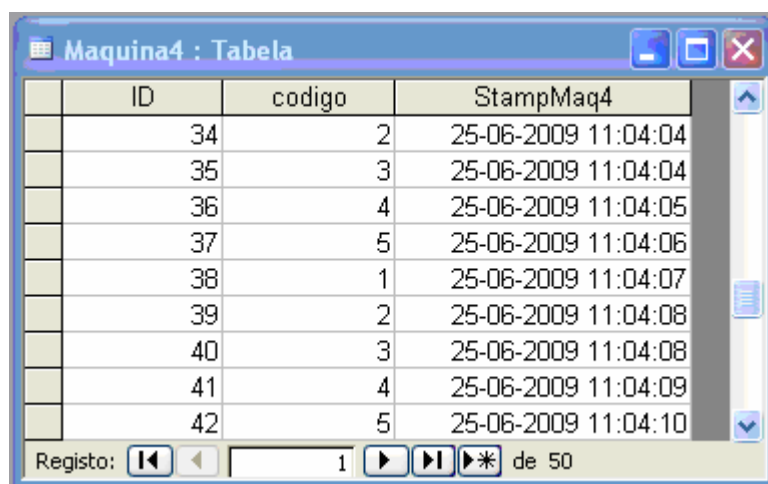
- Execução da conexão e criação de dois grupos: grupo de controlo para verificar o estado da variável de controlo e grupo de dados onde se situam os itens que recolhem a informação proveniente da máquina.
- Activação e desactivação dos grupos, e a leitura por subscrição para o grupo de controlo.
- Visualização do estado da variável de controlo.
- Visualização do código, horas, minutos e segundos do último evento transferido. Estes campos apenas funcionam para ajudar no controlo e acompanhamento dos testes, numa versão final seriam retirados.

Figura 7.9 - Interface do Cliente Síncrono OPC

Descrição do funcionamento

No interface temos duas zonas distintas, “Máquina de Comprovar Portas” e “Leitura e Escrita de Itens”, ambas são autónomas e podem funcionar em simultâneo. Relativamente a zona “Leitura e Escrita de Itens” tudo se processa como na aplicação anterior.

A zona “Máquina de Comprovar Portas” vai servir para a aquisição e registo dos dados provenientes da máquina. Primeiramente selecciona-se o botão “**Conectar**” (quando esta

conectado apresenta a palavra “**Desconectar**”), para registar no servidor os dois grupos. A aplicação emite uma mensagem de erro se não for possível a conexão ao servidor.

Depois desta fase o operador decide quando iniciar a recolha, tendo como possibilidades activar ou desactivar os grupos, desactivar a leitura por subscrição, assim como o registo dos dados na base de dados

Funções relevantes na programação, no que diz respeito ao OPC temos:

- Procura de servidores OPC registados.
- Realização da conexão e desconexão ao servidor OPC escolhido.
- Criação de um ficheiro que guarda as configurações que a aplicação têm ao sair desta.
- Carregamento do ficheiro de configurações ao iniciar a aplicação.
- Cria grupos de itens e adiciona itens a esses grupos.
- Utilização de vários *DataChange* que permitem a leitura por subscrição.
- Activação e desactivação da leitura por subscrição.
- Efectua a leitura síncrona de cada item independente.
- Efectua a escrita síncrona de cada item independente.
- Detecção da falta ou falha do servidor OPC.
- Efectua o tratamento de erros ao nível da conexão e inclusão de itens e carregamento do ficheiro de configurações.
- Realização da sincronização entre o dispositivo e o cliente, recorrendo a tecnologia OPC.
- Introdução do algoritmo que recebe o *time stamp* dos eventos, que é recolhido no relógio de tempo real do autómato em formato BCD, e posteriormente é transferido para três *words*. Seguidamente é decomposto em dia, mês, ano, horas, minutos, segundos para poder ser reagrupado num formato standard que permita cálculos em bases de dados.
- Regista os dados na base de dados.

Na figura seguinte está um excerto da tabela Maquina4 onde podem ser visualizados registos que foram introduzidos pelo cliente na secção “Máquina de Comprovar Portas”.
No campo “código” está o número do evento e no campo “StampMaq4” está o *time stamp* do evento com o instante de tempo recolhido no autómato.

Maquina4 : Tabela

ID	codigo	StampMaq4
34	2	25-06-2009 11:04:04
35	3	25-06-2009 11:04:04
36	4	25-06-2009 11:04:05
37	5	25-06-2009 11:04:06
38	1	25-06-2009 11:04:07
39	2	25-06-2009 11:04:08
40	3	25-06-2009 11:04:08
41	4	25-06-2009 11:04:09
42	5	25-06-2009 11:04:10

Registo: 1 de 50

Figura 7.10 - Imagem da Tabela Maquina4 preenchida pelo cliente OPC

7.5.2 Desenvolvimento de Cliente OPC assíncrono.

O passo seguinte consistiu no desenvolvimento de uma aplicação idêntica em tudo ao cliente síncrono mas que realizasse uma leitura dos itens de forma assíncrona. Ou seja que permitisse que o cliente OPC continue a trabalhar enquanto o servidor recolhe os dados pretendidos. Como já foi referido, para realizar esta função é utilizada uma função de *callback*. O servidor OPC está preparado para aceitar vários pedidos de dados do cliente OPC, e agrupar todas essas solicitações da melhor forma possível para enviar posteriormente ao dispositivo.

O interface é exactamente o mesmo assim como as suas funções apenas é alterado o tipo de leitura.

Enquanto numa leitura síncrona executamos a função "**SyncRead**" e dentro do mesmo procedimento são nos devolvidos os itens, numa leitura assíncrona ao executar a função "**AsyncRead**" apenas é lançado o pedido nesse procedimento, fica activa a partir deste momento uma função de *callback* "**AsyncReadComplet**" que é responsável por devolver os dados pedidos.

A principal dificuldade na implementação deste tipo de leitura, residiu na identificação dos erros que o manual da especificação tinha na inicialização do tipo e dimensão das variáveis. Apesar de ser o tipo de leitura mais eficiente, não será aplicada pelos motivos já descritos em 6.3, dos quais se salienta o paradigma da memória partilhada. No entanto também foram atingidos todos os objectivos nos testes em ambiente real realizados.

7.5.3 Aplicação Final Testada nas Máquinas

Escolhido que está o tipo de leitura de dados, era necessário realizar no cliente síncrono do sub capítulo 7.5.1, algumas alterações finais para realizar o teste na máquina.

Nas aplicações testadas até aqui, foram sempre utilizadas condições reais de utilização, pois tínhamos uma cadeia de dados igual a que seria encontrada na linha, apenas era simulada a ocorrência dos eventos na máquina. Um conjunto de botões simulava a ocorrência de eventos.

Nesta fase foi realizada a configuração da imagem seguinte, directamente na linha e directamente na Máquina de Comprovar Portas, permitindo desta forma testar com total garantia o funcionamento módulo de aquisição de dados.

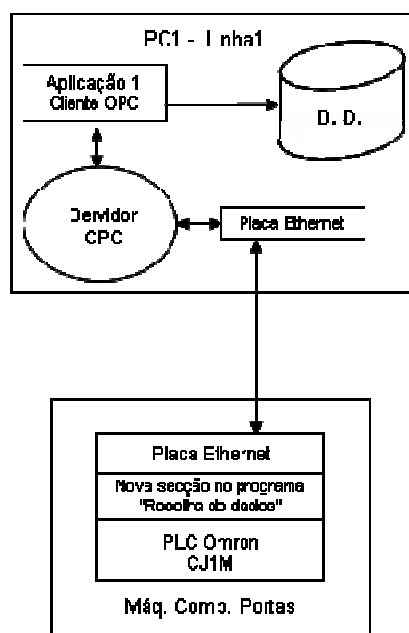



Figura 7.11 – Recolha via OPC

As alterações sofridas passam por:

- Alterar o modelo de registo dos dados na base de dados, passamos a trabalhar com base de dados definida no sub capítulo 7.4.
- Preparar o cliente para trabalhar com varias máquinas ao mesmo tempo, e vários grupos de dados.
- Visualizar último evento transferido. Estes campos apenas funcionam para ajudar no controlo e acompanhamento dos testes, numa versão final seriam retirados.

A aplicação desenvolvida, apresenta o interface da figura seguinte, mantendo as capacidades da aplicação anterior e alterando a secção para visualização do último evento e o número de grupos de dados que pode ler ao mesmo tempo no módulo.

Figura 7.12 – Interface do módulo de aquisição de dados

Descrição de funcionamento

No interface temos duas zonas distintas, “Máquina de Comprovar Portas” e “Leitura e Escrita de Itens”, ambas são autónomas e podem funcionar em simultâneo. Em ambas as zonas tudo se processa como na aplicação anterior.

Funções relevantes na programação, no que diz respeito ao OPC temos:

- Procura de servidores OPC registados.
- Realização da conexão e desconexão ao servidor OPC escolhido.
- Criação de um ficheiro que guarda as configurações que a aplicação têm ao sair desta.
- Carregamento o ficheiro de configurações ao iniciar a aplicação.
- Cria grupos de itens e adiciona itens a esses grupos.
- Utilização de vários *DataChange* que permitem a leitura por subscrição.
- Activação e desactivação da leitura por subscrição.

- Efectua a leitura síncrona de cada item independente.
- Efectua a escrita síncrona de cada item independente.
- Detecção da falta ou falha do servidor OPC.
- Efectua o tratamento de erros ao nível da conexão e inclusão de itens e carregamento do ficheiro de configurações.
- Realização a sincronização entre o dispositivo e o cliente, recorrendo a tecnologia OPC.
- Introdução do algoritmo que recebe o *time stamp* dos eventos, que é recolhido no relógio de tempo real do autómato em formato BCD, e posteriormente é transferido para três *words*. Seguidamente é decomposto em dia, mês, ano, horas, minutos, segundos para poder ser reagrupado num formato standard que permita cálculos em bases de dados.
- Permite funcionar em simultâneo os dois clientes, e vários grupos de dados (várias máquinas).
- Regista os dados na base de dados.

O algoritmo geral do módulo de aquisição de dados para a secção automática de recolha de dados:

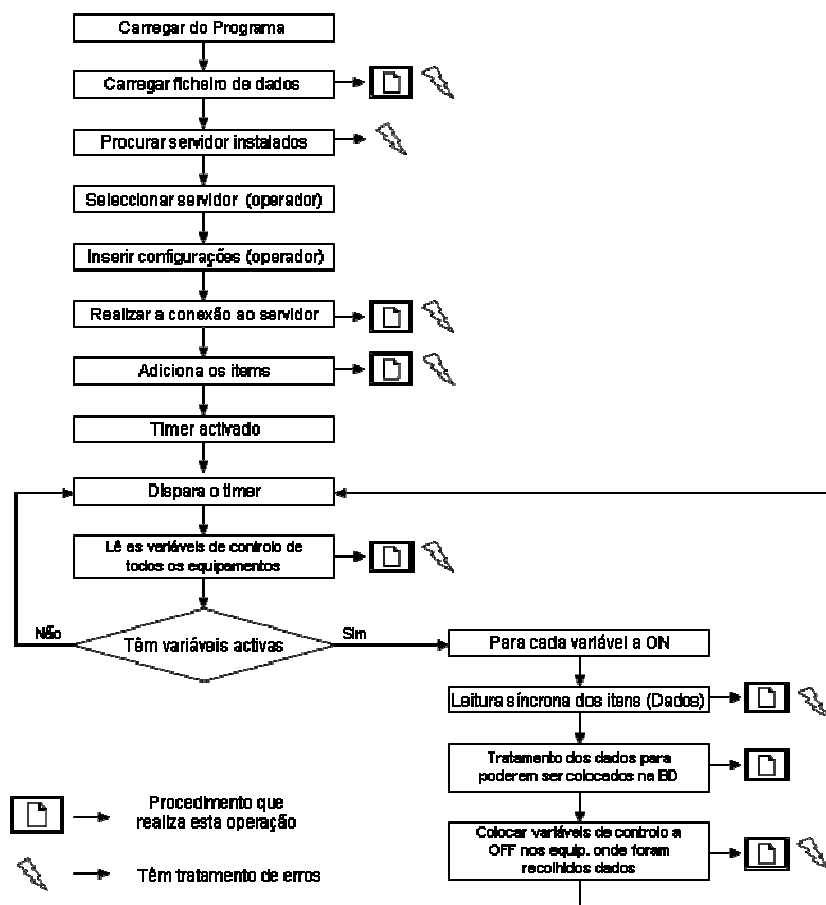



Figura 7.13 - Algoritmo geral do modulo de aquisição de dados

Na figura seguinte está um excerto da tabela “MaqCompPortas” onde podem ser visualizados registos que foram introduzidos pelo módulo durante o funcionamento da máquina de comprovar portas. Os campos estão explicados no sub capítulo 7.4.

MaqCompPortas : Tabela

ID	CodActual	DadosErros	TStamp	DSegundos	Dia1	Mes1	Ano1	TempoCompleto	CodAnterior
6	1	0	16:10:09	1	14	7	2008	14-07-2008 16:10:09	3
7	4	0	16:10:09	4	14	7	2008	14-07-2008 16:10:09	1
8	1	0	16:10:10	4	14	7	2008	14-07-2008 16:10:10	4
9	3	0	16:10:11	4	14	7	2008	14-07-2008 16:10:11	1
10	4	0	16:10:12	4	14	7	2008	14-07-2008 16:10:12	3
11	3	0	16:10:21	3	14	7	2008	14-07-2008 16:10:21	4
12	4	0	16:10:22	2	14	7	2008	14-07-2008 16:10:22	3
13	3	0	16:10:23	1	14	7	2008	14-07-2008 16:10:23	4
14	4	0	16:10:24	1	14	7	2008	14-07-2008 16:10:24	3
15	3	0	16:10:25	0	14	7	2008	14-07-2008 16:10:25	4
16	4	0	16:10:26	8	14	7	2008	14-07-2008 16:10:26	3

Registo: 1 de 40

Figura 7.14 – Imagem da tabela “MaqCompPortas”preenchida num teste na máquina

Para se chegar a versão definitiva foram surgindo problemas que obrigaram a modificações, seguidamente vão ser apresentadas as principais:

Foram corrigidos aspectos no arranque/carregamento do programa, uma vez que se verificava que o facto de a máquina já estar a trabalhar iria provocar em determinadas situações a perda de *buffers* que estavam prontos a ser transferidos.

Outro aspecto que surgiu durante os testes foi, por exemplo uma quebra na linha ou o simples desligar do cabo de rede. Estas situações levaram a que o módulo garantisse que perante um falha deste tipo não haveria perda de informação, desde que o tempo da ocorrência da falha, não ultrapassa-se o tempo gasto para o enchimento dos dois *buffers*.

Um aspecto relevante constatado, reside no facto de o servidor OPC da *Omron*, apenas conseguir tratar um *DataChange*, e ter outro em lista de espera, perdendo os seguintes. (*DataChange* - trata-se de um evento que é disparado quando uma variável tem leitura por subscrição e o valor desta muda)

Este facto levou a alterações significativas no algoritmo do programa, passou-se de um controlo de *Event Driven* para um controlo de *Timer Driven*, ou seja em intervalos de tempo iguais é executado um código que analisa o estado da máquina e a partir dai tomava as decisões. Inicialmente colocou-se a possibilidade de este problema residir no VB6, mas foi desenvolvida uma aplicação dedicada que permitiu concluir que o VB6 era capaz de receber e manter em lista de espera, pelo menos sete eventos. A aplicação desenvolvida permitiu concluir também que o uso de box de mensagens bloqueia os próximos eventos.

Foi ainda utilizado um servidor diferente, este só detectava um novo evento depois de o evento actual ter terminado, esta situação obriga a ter cuidados redobrados na programação e na

utilização de vários servidores, pois o programa pode não trabalhar correctamente com todos eles.

7.6 Aplicação de Teste para o Conjunto da Linha.

Esta aplicação foi desenvolvida a pensar num teste para operar com o conjunto das máquinas da linha, não foi testada directamente na linha, mas cumpriu em testes preliminares todos os objectivos para que foi desenvolvida, no entanto continua em desenvolvimento.

O módulo de aquisição de dados apresentado no sub capítulo 7.5.3 necessita de uns melhoramentos para se parecer em definitivo com uma aplicação final, ou seja tenha um interface mais intuitivo, com mais possibilidade de configurações.

O motor da aplicação é o mesmo mas foram introduzidas funcionalidades como, um ambiente mais intuitivo com menu para selecções tipo *Windows* e possibilidade de procurar e seleccionar servidores em máquinas diferentes (figura 7.15).

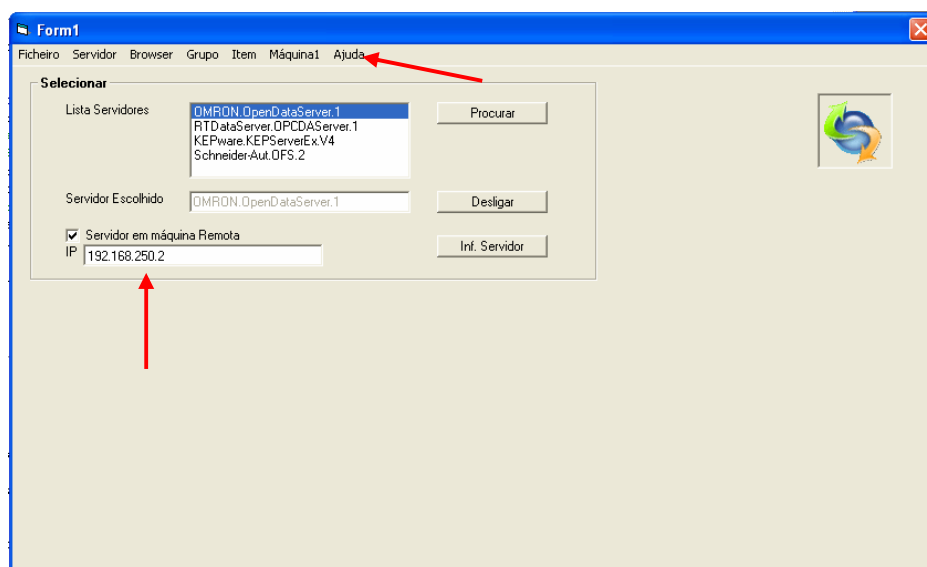



Figura 7.15 – Ambiente gráfico com possibilidade de seleccionar servidor em Maq. Remota

A grande novidade centra-se na inclusão de um **Browser** (figura 7.16) que permite explorar os servidores e escolher os itens desejados para a recolha de dados. Na aplicação anterior era necessário conhecer a localização dos itens no servidor e posteriormente digitá-los. Nesta nova versão, são apresentados todos os itens do servidor numa caixa de selecção.

Figura 7.16 – Interface onde se pode utilizar a ferramenta do Browser

7.7 Resultados e Conclusões

A arquitectura escolhida para o sistema de informação, as configurações nas máquinas e base de dados definida, permitiram atingir, tal como foi demonstrado, os objectivos definidos no ponto 2.4

O módulo de aquisição de dados funcionou e cumpriu com os objectivos pretendidos, tanto na fase de teste em ambiente real, como quando inserido na linha de montagem. De salientar também o facto de se ter revelado fiável, estável e robusto.

Pode-se afirmar com garantia, que este sistema desenvolvido tem capacidade para recolher e registar os eventos relevantes do conjunto de máquinas de uma linha de montagem da *HUF*, ou de uma linha de produção com máquinas que utilizem tecnologias de controlo (máquinas controladas por PLCs e IPCs) idênticas às estudadas neste projecto.

Capítulo 8

Conclusões e Trabalho Futuro

No final de cada capítulo, foram sendo apresentadas as conclusões parciais relativas ao tema aí tratado: Aqui, trata-se de avaliar a satisfação global dos objectivos e apresentar as perspectivas de futuros desenvolvimentos. Serão, no entanto, também revisitados os aspectos mais relevantes do trabalho desenvolvido.

O objectivo inicial de análise e desenvolvimento de um sistema de informação orientado para os processos fabris capaz de (i) recolher automaticamente dados provenientes de um conjunto heterogéneo de máquinas e equipamentos de fabrico e (ii) calcular e apresentar indicadores de desempenho relativos à eficiência dos equipamentos da linha de montagem, com o decorrer do projecto, centraram-se progressivamente no estudo de soluções para a questão da chave da recolha e comunicação de dados provenientes de um conjunto heterogéneo de equipamentos de fabrico.

Neste sentido, começou por ser realizada uma análise ao grupo de máquinas, que apresentava diferentes tecnologias de comunicação, e posteriormente desenvolvido o sistema de informação orientado para o processo em causa, o qual foi organizado nos seguintes módulos:

- Módulo de aquisição de dados - responsável pela recolha e registo de dados subdivide-se em dois sub-módulos. O primeiro opera com as máquinas controladas por PLCs, e é utilizada a tecnologia OPC; o segundo opera as máquinas controladas por IPCs, e é baseado na tecnologia DCOM. Nos dois casos, os dados provenientes dos equipamentos de fabrico são registados em base de dados.
- Módulo de tratamento de dados é responsável pelo tratamento dos dados e pelo cálculo e apresentação dos indicadores de desempenho tais como OEE, tempo de paragem e respectiva causa, tempo para ajustes, faltas de material, tempo de manutenção.

A arquitectura resultante assenta nas tecnologias OPC e DCOM sobre uma rede *ethernet*. Desta combinação resultou uma solução que implementa a comunicação dos dados, de uma forma flexível e que permite futuros desenvolvimentos ou evoluções.

Os testes realizados com a tecnologia OPC permitiram concluir com segurança que o seu desempenho e as suas capacidades são compatíveis com os requisitos desta aplicação, conforme no descrito no capítulo 6. As soluções desenvolvidas têm uma margem de progresso significativa e o uso de algoritmos adequados permitirá a sua utilização em sistemas de grande dimensão. Por seu lado, a tecnologia DCOM também apresenta um desempenho ao nível dos requisitos do projecto. Trata-se, no entanto, de uma tecnologia onde foram surgindo dificuldades na configuração dos clientes e servidores, quando estes foram instalados em máquinas separadas, e ao trabalhar com a *firewall* activa que, no entanto, foram ultrapassadas através de uma cuidada metodologia na sua configuração/instalação, conforme foi descrito no capítulo 4.

Durante o desenvolvimento do módulo de aquisição de dados, foram exploradas e avaliadas várias soluções tecnológicas alternativas. Com efeito, foram testadas várias aplicações que permitiram explorar diferentes soluções para a comunicação de dados, nomeadamente a comunicação síncrona, assíncrona e por subscrição, tendo sido escolhida uma solução intermédia que melhor se adaptava ao projecto e que se baseava numa mistura da comunicação por subscrição com a comunicação síncrona, tal como está descrito em 7.5.

A realização com sucesso dos vários testes do módulo de aquisição de dados, em ambiente fabril real, permite afirmar que os objectivos fundamentais do projecto relativamente à aquisição de dados foram atingidos. Faltou apenas realizar o teste com a máquina de linguetas, pelas razões invocadas no capítulo 4, apesar de tudo estar preparado para a sua realização.

No que diz respeito aos objectivos relativos ao módulo de tratamento de dados, em particular o cálculo dos indicadores de desempenho relativos à eficiência dos equipamentos da linha de montagem não foi integralmente atingido, por ter sido dada prioridade à resolução do problema da aquisição de dados. No entanto, foram já concluídas várias acções relativas do módulo de tratamento de dados como sejam:

- Foi especificada a estrutura e a sua inclusão na arquitectura do sistema de informação já desenvolvido
- Foram desenvolvidas funções de acesso a bases de dados e funções de operações temporais que são uma mais valia em futuros desenvolvimentos, nomeadamente no desenvolvimento do módulo.
- Foram especificados os indicadores de desempenho a calcular bem como a base de dados que vai conter a informação para o seu cálculo.
- Foram realizados testes em condições reais de funcionamento que permitiram adquirir o domínio das tecnologias utilizadas em especial da tecnologia OPC.

Como comentário final, pode afirmar-se que os objectivos propostos inicialmente eram ambiciosos e que, globalmente, o trabalho realizado foi de encontro ao pretendido, dele resultando uma sólida base de trabalho para responder a problemas integração de sistemas.

Por fim, e relativamente, a possíveis desenvolvimentos futuros, apresentamos algumas vias a explorar que complementam o trabalho já desenvolvido:

- Reunir num só computador central o sistema de informação orientado ao processo de todas as linhas de montagem.
- Desenvolver a interface com o utilizador do módulo de tratamento de dados, introduzindo o cálculo de outros indicadores e desenvolvendo um menu de configuração para novas máquinas.
- Realizar um estudo profundo para comparar arquitecturas diferentes, nomeadamente em termo de desempenho entre si, tais como a comparação entre arquitecturas baseadas em redes abertas e redes proprietárias.
- Estudar e comparar o desempenho de aplicações desenvolvidas com diferentes ferramentas de desenvolvimento.
- Desenvolver uma solução que permita reduzir os problemas encontrados, quando temos servidores e clientes em computadores diferentes (configurações DCOM).
- Implementar redundância nas comunicações em especial na tecnologia OPC.

Anexo A

A Apresentação da Empresa Huf

O seguinte anexo serve para apresentar a empresa e ajudar o leitor na interpretação da contextualização, da motivação e dos objectivos da dissertação.

A informação seguinte pode ser consultada em [Huf07].

A.1 Sector de Actividade e Tipo de Produção.

A *Huf* Portuguesa é uma empresa do ramo automóvel, produz sistemas de fechaduras e sistemas de bloqueio de direcção, localizada em Tondela iniciou a sua actividade em Outubro de 1991, e têm como missão "Ser um modelo de excelência no sector".

Figura A.1 - Visão geral da linha de montagem onde foram realizados os trabalhos

Situando-se entre as primeiras cinco maiores empresas do distrito de Viseu apresenta um capital social de 150 000 000€ repartido pelos grupos *Huf* e Ficosa, com um volume de vendas

em 2007 de 98.8 milhões de euros, tem uma cota de mercado na Europa de 22% na produção de sistemas de fechaduras e de 10,8% na produção de sistemas de bloqueio da direcção.

A *Huf* portuguesa trabalha para marcas como Opel, GM, “PSA Peugeot Citroën”, Ford, Jaguar, Land Rover e Grupo Volkswagen, os seus produtos são enviados para os cinco continentes, e pode-se referir que fornece componentes para 12 fábricas da PSA, 9 fábricas da VW, 15 fábricas da GM, 12 fábricas da Ford.

A.2 Áreas e Número de Colaboradores

Com uma área total de 20.000 m^2, dos quais 9.782 m^2 são efectivamente área construída, têm 5.445 m^2 dedicados exclusivamente a produção, 1200 m^2 de armazém e 739 m^2 de área social.

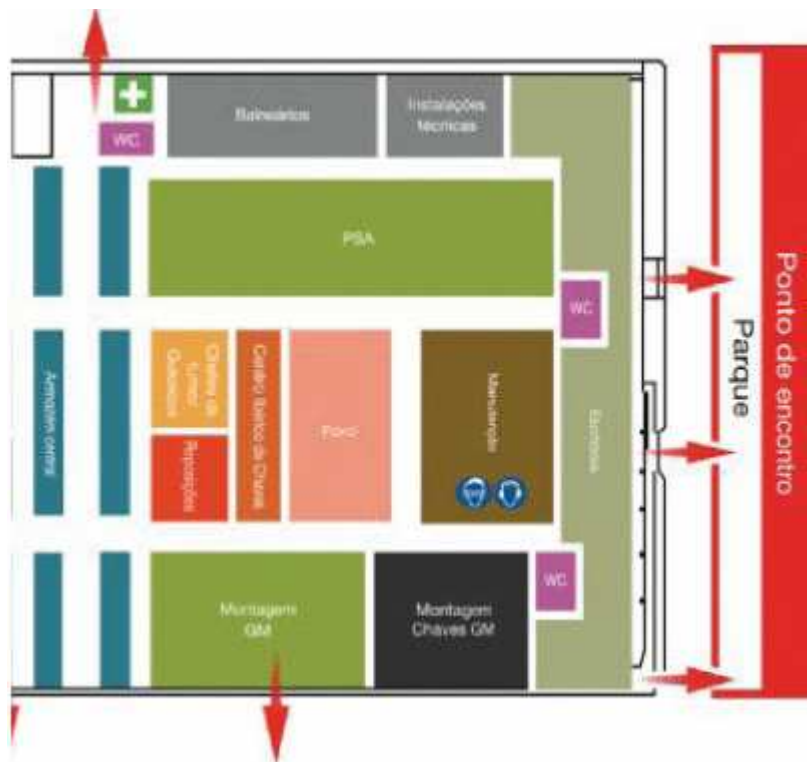



Figura A.2 - Layout da fábrica

Conta com 369 colaboradores repartidos em 51% de Homens e 49% de mulheres.

A.3 Certificações e Prémios

Apresenta uma política de Qualidade/Ambiente/Higiene, Saúde e Segurança ao mais alto nível, a provar são as 7 certificações obtidas até ao momento:

- ISO 9002 no ano de 1996

- QS 9000 no ano de 1997
- VDA 6.1 no ano de 1998
- ISO 14001 no ano de 2000
- ISO 9001: 2000 no ano de 2002
- ISO TS 16949: 2002 no ano de 2003
- OHSAS 18001 no ano de 2003

Figura A.3 - Certificações

E os inúmeros prémios que têm merecido entre os quais se salientam:

Figura A.4 - Prémio Ford Motors - Quality Supplier Award em 1995

Figura A.5 - Prémio GM – Supplier of the year 1998

Premio Melhores Empresas para Trabalhar em Portugal, que foi recebido 4 vezes e que em 2006 ficou num honroso quarto lugar.

Figura A.6 - Premio Melhores Empresas para Trabalhar em Portugal

A.4 Constituição da Empresa

Trabalhando em 3 turnos, a *Huf* conta com diferentes processos produtivos, onde se destacam a injecção de zamak e plástico - com modernos equipamentos de alimentação automática de matéria prima - mecanização e montagem. Em relação à montagem a empresa trabalha em linhas de produção flexíveis que se adaptam às necessidades do mercado, formando células de produção (por produto e modelo) onde se desenvolvem os princípios de Lean Production, Kanban, Melhoria Contínua e Resolução de Problemas (TPM, GAT). O conceito de linha que é utilizado permite uma melhor relação entre produtividade, qualidade e condições de trabalho. Possui uma manutenção que têm como objectivo o eficiente funcionamento de todos os equipamentos através de Acções de Manutenção Preventiva, Preditiva e Correctiva. Com o apoio da engenharia de processos desenvolve máquinas e ferramentas, sistemas anti-erro, sistemas de automatização de processos e sistemas de segurança.

A.5 Funcionamento Geral de uma Linha de Produção.

As linhas de produção são alimentadas por tapetes rolantes que cobrem toda a fábrica, permitindo levar as peças em fase de produção a qualquer local desejado. Nos tapetes rolantes circulam paletes (dimensões aproximadas de 200mm*200mm), que contêm no fim da linha o sistema de fechaduras completo de um automóvel, pronto a ser enviado para o fabricante de automóveis. No início da linha a palete começa a receber componentes que vão sendo agrupados, dando então lugar as fechaduras, no decorrer do processo há também lugar a testes de qualidade.

Os operadores apenas se limitam a seguir o ciclo das mesmas e só intervêm em caso de falha, ou para abastecer as máquinas. Funcionam assim como supervisores ou alimentadores.

A.6 Motivos Porque não Existem Registos Referentes ao Processo de Fabrico

O facto de não existirem registos quantificáveis referentes ao processo fabrico está parcialmente explicada por vários motivos como sejam:

- - Não existia uma necessidade presente de se conhecer estes dados.
- - A empresa apresenta uma sobrecapacidade de produção.
- - Várias máquinas na linha de produção podem compensar-se umas as outras. Na linha de montagem para uma mesma operação de fabrico existem várias máquinas capazes de realizar essa operação. Perante esta situação qualquer paragem ou mesmo avaria de um equipamento, pode ser facilmente minimizada, reorganizando o processo, exceptuando se a máquina for crítica (única na linha de produção, situação diminuta, restrita a uma ou duas operações, mas que podia também ser ultrapassada recorrendo a utilização de uma máquina que realizasse a mesma operação em outra linha de montagem).

Anexo B

B Anomalias Detectáveis na Máquina de Comprovar Peças

Sempre que ocorre uma anomalia aparece uma mensagem de erro na consola, a tabela seguinte apresenta as possíveis anomalias detectáveis pela máquina. Esta tabela foi retirada do manual da máquina.

Tabela B.1 - Tabela de anomalias da máquina de comprovar portas

MENSAGEM	*CAUSA*	*SOLUÇÃO*
ENCRAVAMENTO CILINDRO TRANCA ESQUERDA	Prisão mecânica do cilindro ou desafinação dos detectores	*Verificar cilindro e respectivos detectores*
ENCRAVAMENTO CILINDRO TRANCA DIREITA	Prisão mecânica do cilindro ou desafinação dos detectores	*Verificar cilindro e respectivos detectores*
ENCRAVAMENTO CILINDRO PINÇA	Prisão mecânica do cilindro ou desafinação dos detectores	*Verificar cilindro e respectivos detectores*
ENCRAVAMENTO CILINDRO TRANSPORTE PINÇA	Prisão mecânica do cilindro ou desafinação dos detectores	*Verificar cilindro e respectivos detectores*
ENCRAVAMENTO CILINDRO ROTAÇÃO DE PINÇA	Prisão mecânica do cilindro ou desafinação dos detectores	*Verificar cilindro e respectivos detectores*
ENCRAVAMENTO CILINDRO ELEVADOR DE PINÇA	Prisão mecânica do cilindro ou desafinação dos detectores	*Verificar cilindro e respectivos detectores*
ENCRAVAMENTO CILINDRO ELEVADOR DE PALETE	Prisão mecânica do cilindro ou desafinação dos detectores	*Verificar cilindro e respectivos detectores*

ENCRAVAMENTO CILINDRO 2º AVANÇO DE TRANSPORTE DA PEÇA	Prisão mecânica do cilindro ou desafinação dos detectores	*Verificar cilindro e respectivos detectores*
ENCRAVAMENTO CILINDRO 2º AVANÇO DE ELEVADOR DE PEÇA	Prisão mecânica do cilindro ou desafinação dos detectores	*Verificar cilindro e respectivos detectores*
ERRO - TIPO PEÇA INCORRECTO	A peça não é a que esta programada na consola	*Verificar a peça ou verificar se receita da consola esta bem parametrizada*
ERRO - FALTA PEÇA	Não existe peça no ninho	*Verificar célula*
ALARME - TORQUE MÁXIMO	Célula atingiu valor máximo programado	*Verificar peça ou prisão*
ERRO TORQUE - RODAR A ESQUERDA	Valor de torque programado atingido	*Verificar a peça*
ERRO TORQUE - FALTA DETECÇÃO MOLA	Valor de torque programado atingido	*Verificar a peça*
ERRO TORQUE - RODAR A DIREITA	Valor de torque programado atingido	*Verificar a peça*
ERRO - SERVO MOTOR OMRON	Servo não atingiu a cota de destino	*Verificar origem da prisão do motor*
ERRO – ALTURA (CE1) MESA POSICIONAMENTO	Cota da altura da peça diferente da cota programada na receita	*Verificar peça ou corrigir valor da receita*
FALHA DE PRESSÃO DE AR	Pressão de alimentação de ar esta abaixo da pressão programada	*Corrigir a pressão de ar*
PORTAS ABERTAS	Portas traseiras abertas	*Fechar as portas*
FALTA REFERENCIAR MESA SMC	Mesa da SMC não esta referenciada	*Pressionar no Botão de RESET 2 Segundos*
FALTA REFERENCIAR SERVO MOTOR	Motor não esta referenciado	*Pressionar no Botão de RESET 2 Segundos*

Anexo C

C Resultados do Estudo de Performance Realizado pela OPC Foudation

Os resultados do estudo foram os seguintes:

1. Mínimo *Update Rate* dependendo do numero de grupos (cliente e servidor no mesmo PC / carga de 5%).

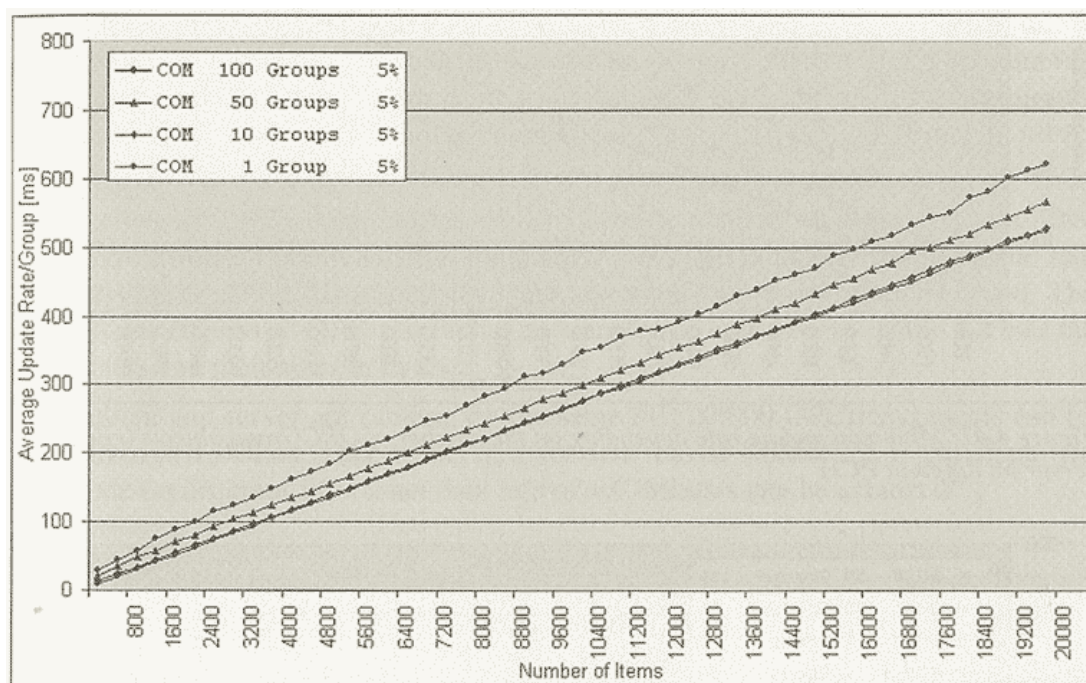



Figura C.1 - Mínimo Update Rate dependendo do numero de grupos (COM)

2. Mínimo *Update Rate* dependendo do numero de grupos (cliente e servidor em PCs separados / carga de 5%).

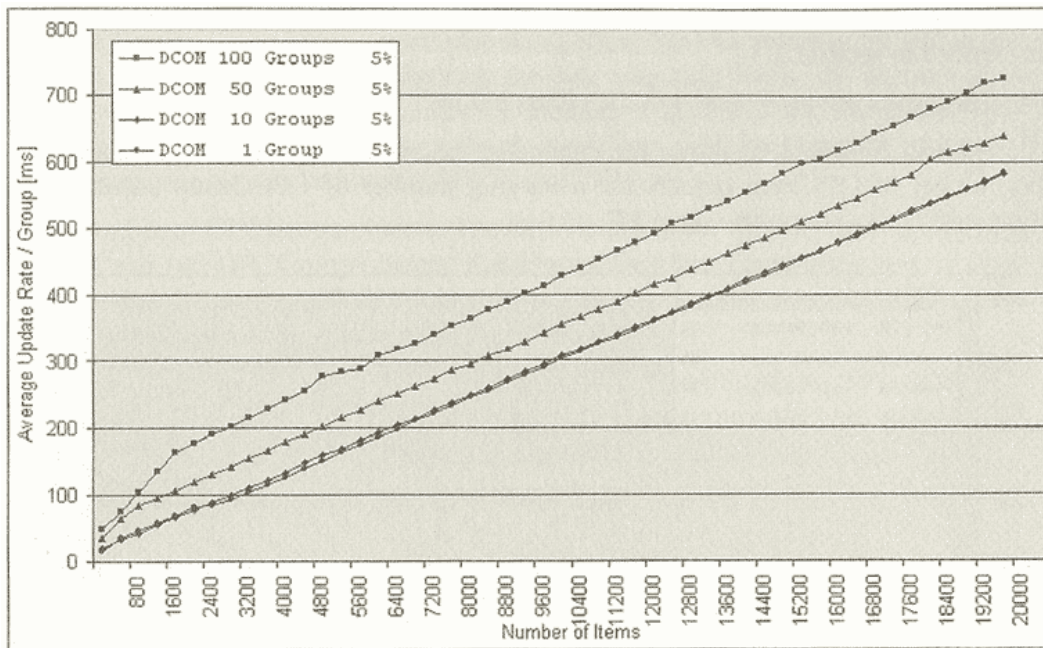



Figura C.2 - Mínimo Update Rate dependendo do numero de grupos (DCOM)

3. Comparação do mínimo *Update Rate* dependendo do numero de grupos para o servidor e cliente na mesma máquina e em máquinas separadas.

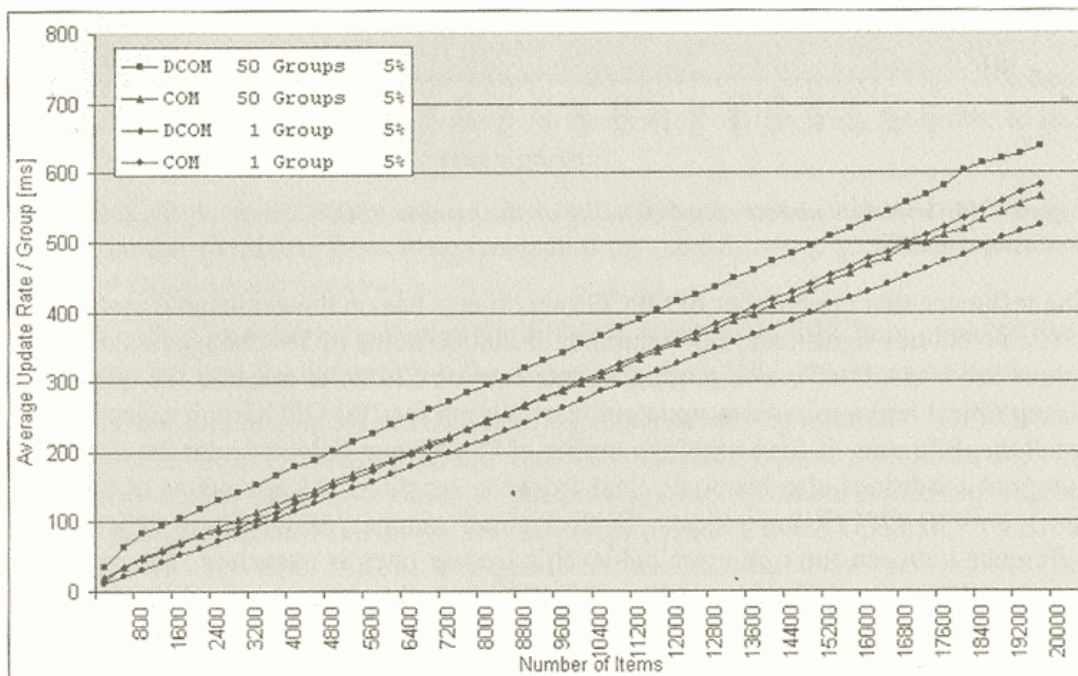



Figura C.3 - Comparação do mínimo Update Rate dependendo do numero de grupos

4. Mínimo *Update Rate* dependendo de diferentes taxas de variação dos itens (cliente e servidor no mesmo PC).

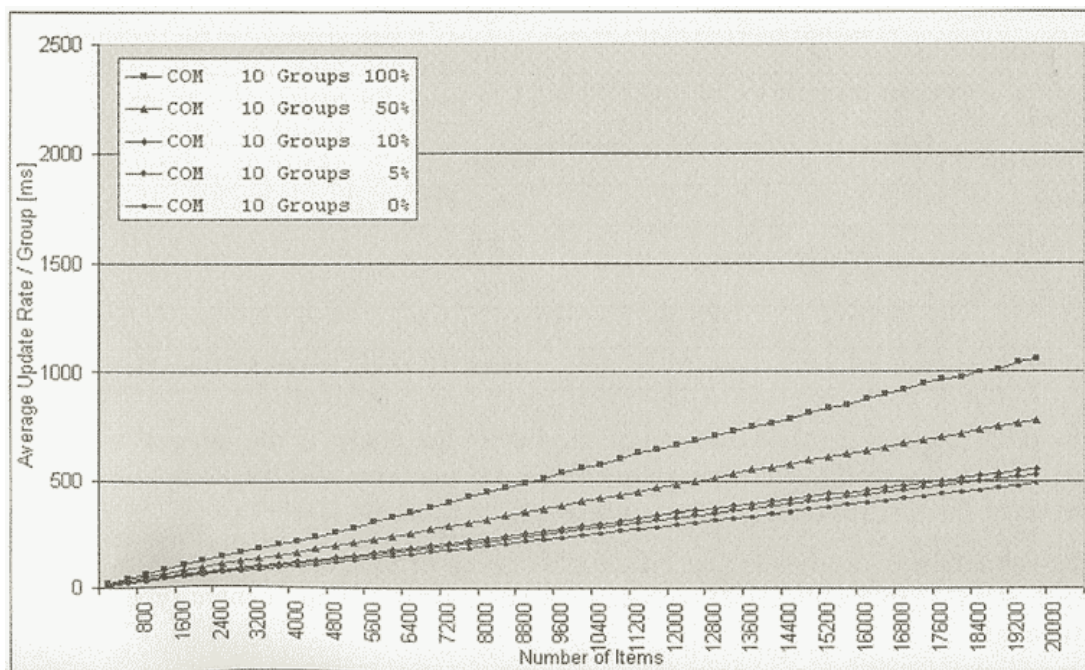



Figura C.4 - Mínimo Update Rate dependendo de diferentes taxas de variação dos itens (COM)

5. Mínimo *Update Rate* dependendo de diferentes taxas de variação dos itens (cliente e servidor em PCs separados).

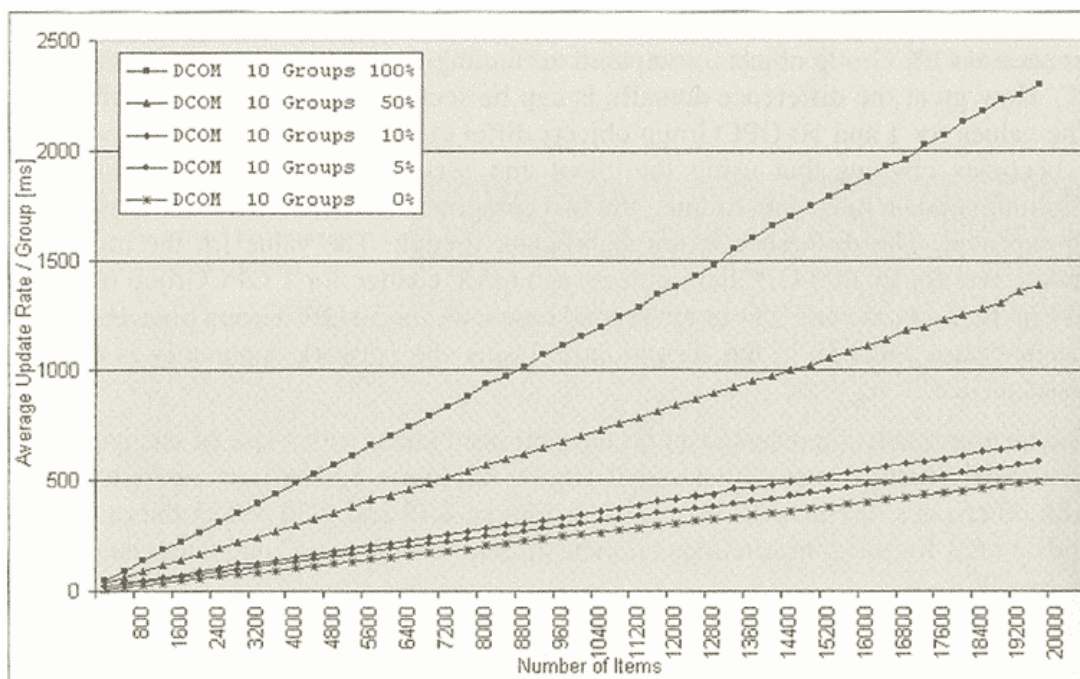



Figura C.5 - Mínimo Update Rate dependendo de diferentes taxas de variação dos itens (Dcom)

6. Comparação do mínimo *Update Rate* dependendo de diferentes taxas de variação dos itens (servidor e cliente na mesma máquina e em máquinas separadas).

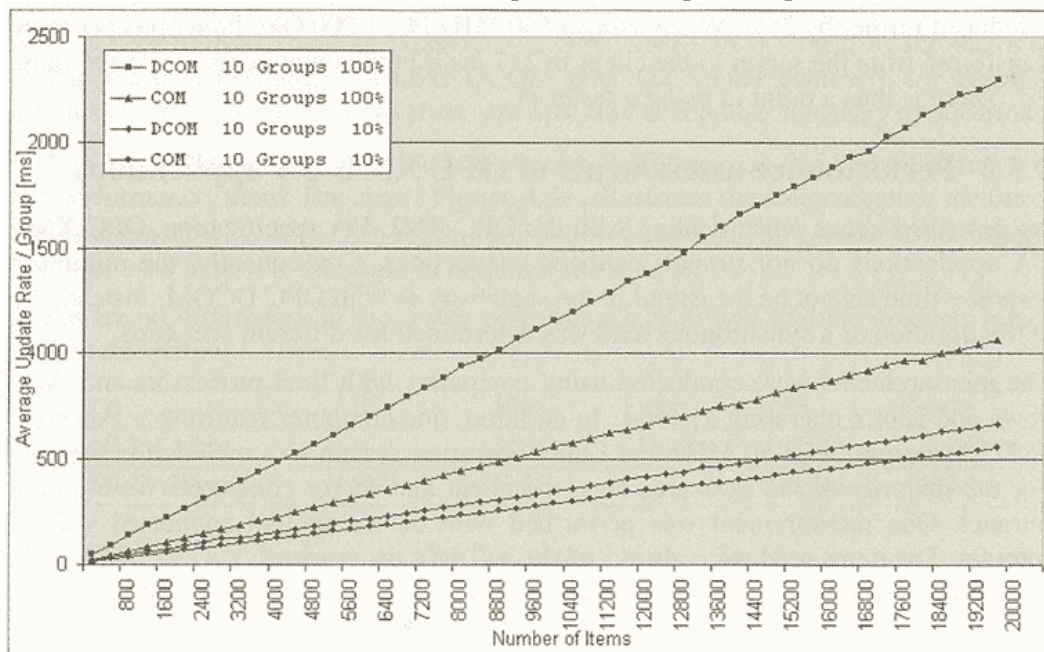



Figura C.6 - Comparação do mínimo Update Rate dependendo de diferentes taxas de variação dos itens

7. Mínimo *Update Rate* dependendo da velocidade do processador.

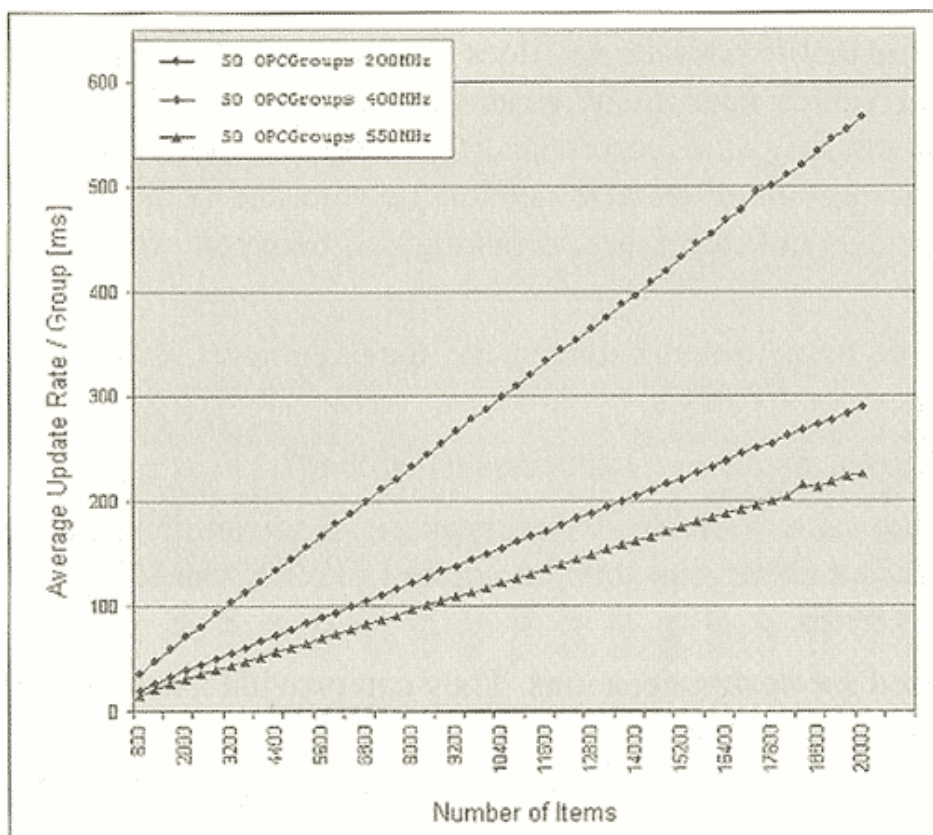



Figura C.7 - Mínimo Update Rate dependendo da velocidade do processador

Referências

[Iwa06] Iwanitz. Frank; Lange. Jürgen, “OPC-Fundamentals, Implementation, and Application”, Hüthig , third edition, Germany, 2006. ISBN 3-7785-2904-8.

[OPC08a] About OPC, disponível em http://www.opcfoundation.org/SiteMap.aspx?MID= AboutOPC, Acesso em Maio de 2008.

[OPC08b] “OPC Overview”, disponível em http://www.opcfoundation.org/Archive/c9cf0fff-65fa-4362-bc93-aa7d0ab58016/General/OPC%20Overview%201.00.pdf, Acesso em Maio de 2008.

[OPC08c] “Data Access Automation Interface Standard”, disponível em http://www.opcfoundation.org/DownloadFile.aspx?CM=3&RI=68&CN=KEY&CI=283&CU=4, Acesso em Maio de 2008.

[OPC08d] Specifications, disponível em http://www.opcfoundation.org/SiteMap.aspx?MID= Downloads, Acesso em Maio de 2008.

[OPC09e] “OPC UA, Part 1: Overview and Concepts”, disponível em https://www.opcfoundation.org/DownloadFile.aspx?CM=1&RI=414, Acesso em Fevereiro de 2009.

[Pau05] Portugal. Paulo, “OPC - Open Process Control, Tecnologias de Automação 2005/06”, FEUP, 2005.

[Pau06] Portugal. Paulo, “OPC DA, Tecnologias de Automação 2006/07”, FEUP, 2006.

[Kir08a] H. Kirrmann, EPFL, ABB Research Center, Baden, Switzerland, "OLE for process Control (OPC), Common elements "disponível em http://lamspeople.epfl.ch/ kirrmann/Slides/AI_431_OPC_Common.ppt, Acesso em Maio de 2008.

[Kir08b] H. Kirrmann, EPFL, ABB Research Center, Baden, Switzerland, "OLE for process Control (OPC), Data Access Specification" http://lamspeople.epfl.ch/kirrmann/Slides/AI_432_OPCDA.ppt, Acesso em Maio de 2008.

[Adr08] Pedroso Puda. Adriano, "Padronização da Comunicação através do Tecnologia OPC", ISA – Sociedade de Instrumentação Sistemas e Automação, disponível em http://www.isarj.org.br/artigos/Padronizacao-da-Comunicacao-atraves-da-Tecnologia-OPC.pdf, acesso em Maio de 2008.

[Jos06] António Faria. José, "Programação de Sistemas de Bases de Dados", FEUP / DEEC, 2006.

[Huf07] "Universo da Empresa – Informação Institucional", *Huf* Portuguesa, Fevereiro de 2008.

[Mig09] Monteiro. Miguel Pimenta, "Algumas notas Sobre COM e DCOM, Tecnologias de Distribuição e Integração 2007/08", FEUP, 2009

[Com09] What is COM?, disponível em http://www.microsoft.com/com/default.mspx, Acesso em Fevereiro de 2009.

[Dcn09] DCOM Technical Overview, disponível em http://msdn.microsoft.com/en-us/library/ms809340.aspx, Acesso em Fevereiro de 2009.

[Dcn09b] DCOM Architecture, disponível em http://msdn.microsoft.com/en-us/library/ms809311.aspx, Acesso em Fevereiro de 2009.

[Tol09] Toledo, Maria Beatriz Felgar de, "COM/DCOM" Instituto de Computação – Unicamp, disponível em http://www.ic.unicamp.br/~beatriz/cursos/mo809/2006/ slides/COM_DCOM.pdf, Acesso em Fevereiro de 2009.